Titre: Title:	Anesthésie robotique : insertion automatique d'un cathéter veineux guidée par ultrasons
Auteur: Author:	Germain Aoun
Date:	2010
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Aoun, G. (2010). Anesthésie robotique : insertion automatique d'un cathéter veineux guidée par ultrasons [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/378/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/378/
Directeurs de recherche: Advisors:	Mohamad Sawan, & Thomas Hemmerling
Programme: Program:	Génie biomédical

UNIVERSITÉ DE MONTRÉAL

ANESTHÉSIE ROBOTIQUE : INSERTION AUTOMATIQUE D'UN CATHÉTER VEINEUX GUIDÉE PAR ULTRASONS

GERMAIN AOUN

INSTITUT DE GÉNIE BIOMÉDICAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION

DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES

(GÉNIE BIOMÉDICAL)

JUILLET 2010

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée:

ANESTHÉSIE ROBOTIQUE : INSERTION AUTOMATIQUE D'UN CATHÉTER VEINEUX GUIDÉE PAR ULTRASONS

Présenté par : AOUN GERMAIN

en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

M. VINET ALAIN, Ph. D., président

M. SAWAN MOHAMAD, Ph. D., membre et directeur de recherche

M. HEMMERLING THOMAS, M.D., membre et codirecteur de recherche

M. SINHA AVINASH, M.D., membre

REMERCIEMENTS

J'offre mes remerciements les plus sincères à mes directeurs de recherche, le Docteur Thomas M. Hemmerling et le professeur Mohamad A. Sawan. Ce travail fut d'autant plus agréable grâce à leurs conseils qui ont permis d'en améliorer la qualité.

Je remercie le professeur Alain Vinet et le Docteur Avinash Sinha qui ont accepté d'évaluer mon mémoire et de faire partie du jury.

Je tiens aussi à remercier chaleureusement toutes les personnes qui ont collaboré à ce projet, notamment les membres de l'équipe du laboratoire Intelligent Technologies in Anesthesia Group (ITAG) : Samer Charabati, Roy Kazan et Nhien Lê.

Finalement, j'offre mes remerciements à tous ceux qui ont cru en moi et qui n'ont cessé les encouragements tout au long de cette longue période.

RÉSUMÉ

—**But**: L'automation est un concept qui a révolutionné plusieurs domaines, allant de l'industrie automobile aux télécommunications, en passant par la médecine. Cependant, ce concept est quasi-inexistant en anesthésie. Ce projet vise à développer un système automatique pour une insertion intraveineuse d'un cathéter, et donc à déclencher l'automatisme dans le domaine de l'anesthésie.

—**Méthodologie**: Un bras robotique effectue un balayage d'une sonde ultrasonique sur un bras fantôme (contenant une veine artificielle). Cette vidéo est transférée à un ordinateur et des images fixes en sont analysées par des algorithmes Matlab. La veine est détectée et ses coordonnées sont envoyées à un second bras robotique qui positionne une aiguille et un cathéter au-dessus du point d'insertion.

—**Résultats**: Deux bras robotiques sont construits: Le premier, guidé par 6 servomoteurs, effectue un balayage d'une sonde ultrasonique sur le site veineux; et le second, guidé par 7 servomoteurs, positionne le cathéter et l'aiguille au point d'insertion, puis y insère l'aiguille en premier et le cathéter en second. L'analyse vidéo et la détection de veine sont faites à travers une interface graphique Matlab, qui utilise la transformée de Hough.

Mots clé — Anesthésie, Robotique, Ultrasons, Accès veineux, Insertion d'aiguille

ABSTRACT

—**Purpose**: Automation is a common concept that revolutionized multiple domains, from automobile industry to telecommunications, passing by medicine. Little automation has been introduced to the anesthesia domain. The purpose of this project is to develop an automated system for intravenous catheter insertion, and thus trigger automation use in the anesthesia domain.

—**Methods**: A robotic arm performs ultrasound scans on the venous site (a phantom arm), real-time snapshots are analyzed by Matlab algorithms, the vein is detected, and corresponding coordinates are sent to a second robotic arm that will position the catheter and needle bundle at the correct insertion point.

—**Results**: Two robotic arms were built: the first one, guided by 6 servomotors, performs automatic ultrasound scans on the venous site, and the second one, guided by 7 servomotors, positions the catheter and needle at insertion point, then inserts the needle in the vein. Analysis is performed on a Matlab graphical user interface (GUI), using Hough transform.

Index Terms—Anesthesia, Robotics, Ultrasound, Image-guided, Venous access, Needle insertion

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ANNEXES

INTRODUCTION

En anesthésie, l'administration de drogues anesthésiantes se fait par l'insertion de l'aiguille d'une seringue dans des veines de bras ou de la main. Cette tâche d'insertion veineuse est généralement effectuée par des personnes appartenant au corps hospitalier, mais pas nécessairement au corps médical, tels que les inhalothérapeutes, infirmiers, paramédicaux, etc. Ceux-ci ne garantissent pas la même qualité d'insertion qu'un médecin ou anesthésiste. De plus, les facteurs de risque (infections ou autre) présents pour un personnel non-qualifié sont assez élevés et la sécurité du personnel hospitalier et du patient est mise en jeu. L'utilisation d'appareils à ultrasons permet de visualiser les veines lors des ponctions et offre une meilleure qualité d'insertion de veine, et ainsi une meilleure qualité de soins au patient.

Ainsi, et à la base de ces idées, le présent projet a été conçu pour automatiser cette partie initiale de l'anesthésie et offrir à l'anesthésiste le temps nécessaire pour se consacrer aux taches importantes et faire un suivi de l'état du patient. Ainsi, la veine sera détectée par une analyse d'images enregistrées par des ultrasons pour transmettre les coordonnées correspondantes à un système facilitant l'insertion automatique de l'aiguille de la seringue dans la veine visée. Par conséquent, les erreurs dues à la mauvaise détection visuelle de la veine et à une mauvaise insertion conséquente de l'aiguille seront largement minimisées et le processus manuel remplacé par un système automatique.

L'insertion automatique de l'accès veineux, guidée par ultrasons, garantit ainsi la sécurité du patient et la qualité du soin qui lui est fourni.

1.1 Anesthésie

L'anesthésie est l'acte qui consiste à endormir et à rendre insensible le patient (anesthésie générale) ou une partie de son corps (anesthésie locale). Ceci permet aux patients, en cours de chirurgie, d'avoir une suppression de la sensation de douleur **[1]**. Le fait de rendre le patient, ou une partie de son corps, insensible existe depuis les temps préhistoriques. Des fouilles archéologiques montrent que des capsules d'opium étaient utilisées à ces fins aux alentours de 4200 BC **[2]**. Ce savoir se répandit d'empire en empire, du peuple Sumérien, à l'empire Chinois, aux Grecs, aux Perses et Arabes, pour enfin arriver aux Amériques. Même avec ces connaissances de base, la plupart des chirurgies connues de nos jours ne pouvaient encore être effectuées. En 1842, Dr. William Clarke utilisa l'éther pour extraire une dent d'un patient. La même année, Dr. Crawford Long performa la première chirurgie sans douleur, en utilisant l'éther aussi. En 1844, Dr. Horace Wells introduit l'oxyde nitreux en dentisterie et remarqua que l'inhalation de ce gaz (aujourd'hui connu sous le nom de « gaz hilarant ») produit un état d'intoxication durant lequel les patients devenaient insensibles à la douleur. Les travaux de Dr. Clarke, Dr. Long et Dr. Wells ne furent rendus publics que plus tard. Il a fallu attendre Dr. William Morton (Figure 1-1), un dentiste de Boston, fut le premier à pratiquer une anesthésie proprement dite. Il utilisa l'éther et tenta de cacher la véritable nature de l'anesthésiant en le nommant « Letheon » et en brevetant la substance.

Figure 1-1. William Morton, dentiste de Boston, a effectué la première anesthésie en 1846 **[3]**

Bien que cette technique existe depuis des centaines d'années, sous forme de divers élixirs soporifiques, tels le suc de pavot, le chanvre, la mandragore, l'opium et autres, le mot « anesthésie » ne fut utilisé pour la première fois qu'en 1846 par Sir Oliver Wendell Holmes, dans une lettre adressée à Dr. William Morton. Sir Holmes dériva le mot « *anæsthesia* » du latin « an » (→ priver) et « aïsthêsis » (→ sensibilité) **[2]**. Depuis, divers agents anesthésiants furent essayés, tels que le chloroforme, la cocaïne, les opioïdes, etc., surtout après la seconde guerre mondiale, durant laquelle plusieurs développements eurent lieu pour entrouvrir de nouveaux horizons dans le domaine médical en général, et en anesthésie en particulier.

Ainsi, l'anesthésie, telle que connue de nos jours, peut être répartie en quatre catégories **[4]** :

- Sédation minimale : Les fonctions cognitives et de coordination du patient sont bloquées, mais celui-ci peut répondre aux commandes verbales et ses systèmes cardiovasculaire et respiratoire ne sont pas affectés
- Sédation modérée : Le patient a besoin d'aide pour effectuer un mouvement. Aucune intervention n'est nécessaire pour les systèmes cardiovasculaire et respiratoire.

- Sédation profonde : Le patient répond seulement aux stimulations de douleurs (réflexes). Les fonctions cardiovasculaires sont maintenues, alors que le support est nécessaire au maintien des fonctions respiratoires.
- Anesthésie générale : Le patient ne répond à aucun stimulus (volontaire ou réflexe) et est incapable de respirer sans l'aide d'un respirateur. Le système cardiovasculaire peut être affecté. L'anesthésie générale est donc une perte volontaire et réversible de la conscience et de toute sensation dans un but thérapeutique. Elle consiste en trois composantes, soit l'état d'hypnose (perte de conscience), l'analgésie (insensibilité à la douleur), et blocage neuromusculaire (perte de réflexes).

1.2 Concept d'automatisme

Les termes « automate », « androïde », « cyborg », ou « robot » ramènent des images de R2D2 et C-3P0 de « La Guerre des Etoiles » (version française de « Star Wars »), des récentes innovations japonaises telles que « ASIMO » (androïde humain, de Honda), « Aibo » (chien robot de compagnie, de Sony) et plein d'autres similaires. Mais, en fait, l'idée ne doit pas stagner à ce concept, car tout mouvement séquentiel, simple ou multiple, effectué par une machine ou groupe de machines pour réaliser une suite de tâches **[5]** peut être désigné comme étant un automatisme. Techniquement, un automatisme est un sous-ensemble de machine(s) destiné à remplacer de façon « automatisée » une action ou décision habituelle et prédéfinie sans intervention de l'être humain. Ces machines peuvent donc produire un mouvement prédéterminé, fixe, ou même « intelligent ».

On peut donc admettre que, généralement, un automatisme peut être subdivisé en deux sous-ensembles :

- Un organe de décision, où les commandes sont prises
- Un organe d'action, où les commandes précédemment prises sont effectuées. Cette partie opérative peut être mécanique, électrique, pneumatique ou hydraulique et bien souvent un assemblage de ces technologies.

L'interaction entre ces deux organes peut être unilatérale ou bilatérale. Ainsi, dans certains cas, l'organe de décision peut envoyer les commandes à l'organe d'action sans que ce dernier ne renvoie de réponse. Alors que dans d'autre cas, l'organe d'action retransmet à l'organe de décision un compte rendu de l'action qui a été effectuée pour que les nouvelles commandes en soient affectées.

Sur ces bases, le concept d'automatisme, proprement dit, s'est ainsi développé dans différents domaines tels que **[5]** :

- Les services financiers (ATM, cartes de crédits, etc.)
- Les services de gouvernement (scanneurs optiques de postes, distribution de courant, etc.)
- Les services de communication (répondeurs automatiques, téléconférence, etc.)
- Les services de transport (contrôle aérien, autopilotage, etc.)
- Les services d'éducation (ordinateurs, calculatrices, audio-visuels, etc.)
- Les services de restaurants (assemblage, scanners optiques, etc.)
- Les services d'habitat (réservations électroniques, ascenseurs, etc.)
- Les services d'amusement (jeux télévisés, projecteurs vidéo, etc.)
- Et, enfin, les services de santé (stimulateurs cardiaques, CAT scans, tables d'opération, etc.)

Nous pouvons aussi, de nos jours, retrouver pleins d'automatismes autour de nous. De la machine à laver le linge, à la vaisselle, au réveil matin, aux robots industriels qui effectuent les tâches humainement impossibles ou dangereuses, aux stimulateurs cardiaques, et bien d'autres. Tous aident l'être humain à réaliser des actions répétitives, précises, rapides, et sécuritaires.

1.3 Évolution de l'interface avec l'utilisateur

Les premiers développeurs considéraient les interfaces graphiques comme étant inutiles et étaient plus intéressés par le développement du processeur et du nombre de computations effectuées par les premiers ordinateurs.

Avec les avancements technologiques des années 60-70, quelques efforts furent exploités pour optimiser les temps d'entrées de données et d'interaction humain-machine. En effet, avec les réductions des coûts liés au prix et à la puissance des processeurs, des compagnies telles que IBM et Xerox concentrèrent leurs efforts sur le développement d'interface, ce qui résulta en la création du centre de recherche de Palo Alto (Palo Alto Research Center – PARC). L'équipe PARC se distingua par plusieurs nouveautés, notamment l'usage de la souris, les systèmes de réseautage, les espaces communs, et les premières interactions facilitées entre humain et machine.

A travers des études cognitives psychologiques, icônes et images graphiques furent introduites, au fur et à mesure, dans ces interactions, engendrant ainsi les premiers battements d'ailes des interfaces graphiques utilisateurs (Graphical User Interface – GUI).

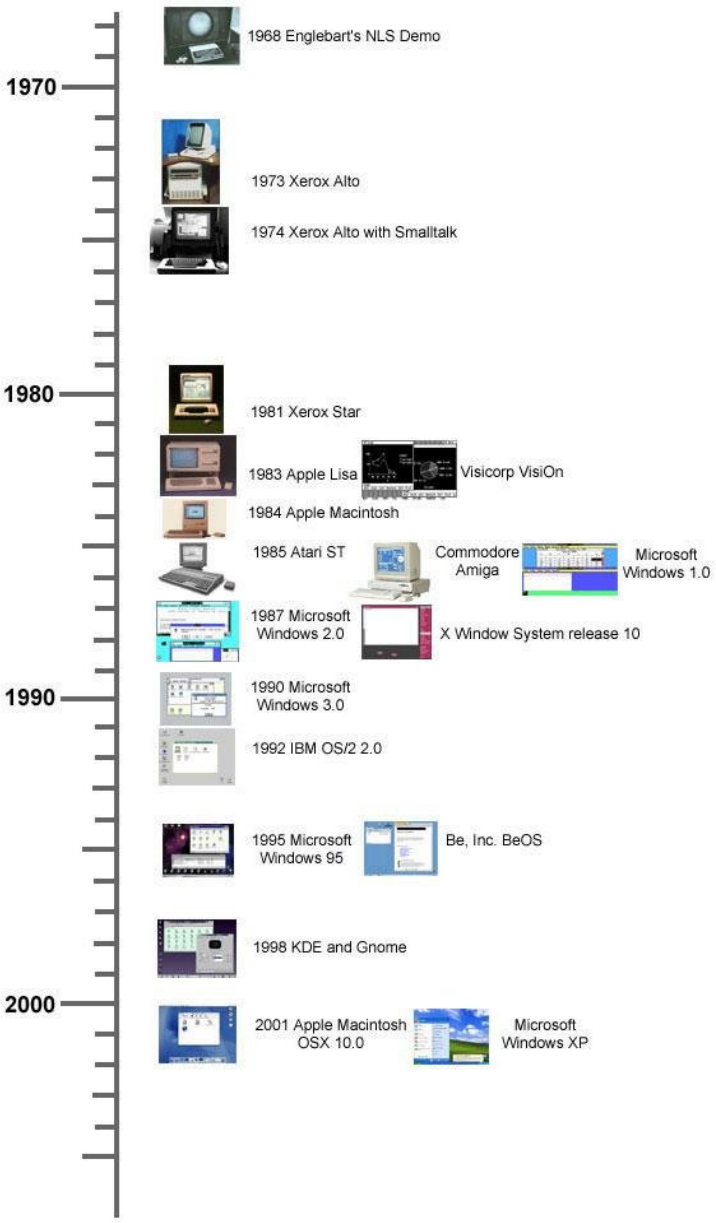


Figure 1-2. Évolution des interfaces graphiques [6]

Ainsi, l'évolution logique eut lieu, et les interfaces évoluèrent (Figure 1-2) du simple DOS, au NLS en version démo (Englebart, 1968), au PARC Alto (Xerox, 1973), au Macintosh (Apple 1984) au Windows 1.0 (Microsoft, 1986) pour en arriver aux plus récents et complexes, tels que Mac OS X et Windows Vista. **[7]**.

Bien que le développement des interfaces graphiques a été principalement associé au monde des ordinateurs, cette idéologie a aussi été élaborée dans d'autres technologies, telles que les cellulaires, les équipements médicaux, les interfaces bancaires, etc. De nos jours, cette technologie est couramment utilisée dans les domaines à haut risque, tels le monitorage des centrales nucléaires, l'aviation, les voyages dans l'espace, etc. Avec l'évolution des interfaces graphiques, la productivité a donc considérablement augmenté, puisque les utilisateurs ne sont plus dépendants de la compréhension du jargon informatique, mais plutôt d'un médium graphique et facile à manipuler.

1.4 Objectif du projet

L'objectif principal de ce projet est de développer un système automatique qui effectue une analyse tridimensionnelle d'images ultrasons afin d'assurer la détection automatique d'une veine dans un espace 3D prédéfini et d'en ressortir ses coordonnées. Ce système contrôle un bras robotique articulé par des moteurs qui assurent avec précision le mouvement d'une seringue qui permettra l'insertion d'un cathéter dans la veine préalablement détectée pour y injecter des agents anesthésiants.

Ainsi, cet objectif est réparti en sous-objectifs :

1. Développement d'un système mécanique pour aider une sonde ultrasons à balayer un site prédéfini pour y détecter une veine
2. Élaboration d'une interface avec l'utilisateur pour surveiller le bon fonctionnement du traitement et analyse d'information
3. Développement d'un système mécanique pour placer l'aiguille et le cathéter sur la veine détectée
4. Insertion de l'aiguille puis du cathéter dans la veine

1.5 Hypothèses de recherche

Nous pouvons développer un système qui permet la détection automatique d'une veine.

Un balayage du site à étudier (généralement le bras) a lieu à l'aide d'une sonde d'ultrasonographie. Une vidéo est générée et analysée par un algorithme. Ceci permet le calcul des coordonnées de la veine par rapport à un espace 3-D.

Ce système permet aussi l'insertion automatique d'une aiguille, puis d'un cathéter dans la veine détectée.

En se basant sur les coordonnées, un bras mécanique positionne un système aiguille-cathéter correctement au-dessus de la veine. L'aiguille perce la peau, puis la veine, pour ensuite permettre au cathéter de se glisser dans la veine et y injecter l'agent anesthésiant.

CHAPITRE 2 REVUE DE LITTÉRATURE

2.1 Automatisme en médecine

Remplacer ce texte Bien que l'aspect technologique ait fait son apparition dans le domaine médical depuis déjà plusieurs années, ce n'est que récemment que quelques visionnaires ont proposé des concepts d'automatisme, ou de robotique, pour diverses applications médicales **[8]**. La première expérience médicale « robotique » fut enregistrée en 1985; il s'agissait d'un dispositif manœuvrant une aiguille pour une biopsie de cerveau **[9]**. Cependant, bien que la première utilisation d'un « robot », date de plus de 20 ans, le domaine de robotique médicale est encore un domaine émergeant, loin d'atteindre sa capacité optimale, et le nombre de systèmes automatiques utilisés en médecine est encore à très petite échelle **[10]**. Malgré cette présence minime, la course à la robotisation du domaine médical est devenue aujourd'hui un des enjeux principaux des marchés économiques mondiaux. En effet, contrairement aux être humains, ces systèmes automatiques peuvent répéter inlassablement des mouvements de façon précise, rapide et sans fatigue, soulageant ainsi l'être humain de tâches répétitives et lui permettant de dépenser son énergie sur d'autres tâches plus importantes **[11]**.

De plus, les erreurs humaines dans le domaine médical n'ont fait qu'augmenter au cours des dernières décennies, nuisant aux patients et augmentant les coûts divers **[12]**. Ainsi, en 1993 seulement, des estimations de 7000 morts ont été liées aux erreurs humaines aux États-Unis avec des coûts totaux excédant les deux milliards de dollars **[13]**.

Figure 2-1. « acCellerator », système automatique de culture de cellules **[11]**

Le but n'est donc pas de remplacer l'être humain et d'éliminer des opportunités de travail, mais plutôt de l'aider dans ses tâches. Ainsi, et avec l'importance croissante de ces systèmes, la course à l'automatisation est devenue tellement rapide qu'il est impossible de citer toutes les applications possibles. Il est même impossible de les classifier par degré de manipulation, degré d'autonomie, d'environnement d'opération ou de but visé **[14]**.

Il serait ainsi préférable de subdiviser la médecine robotique comme suit :

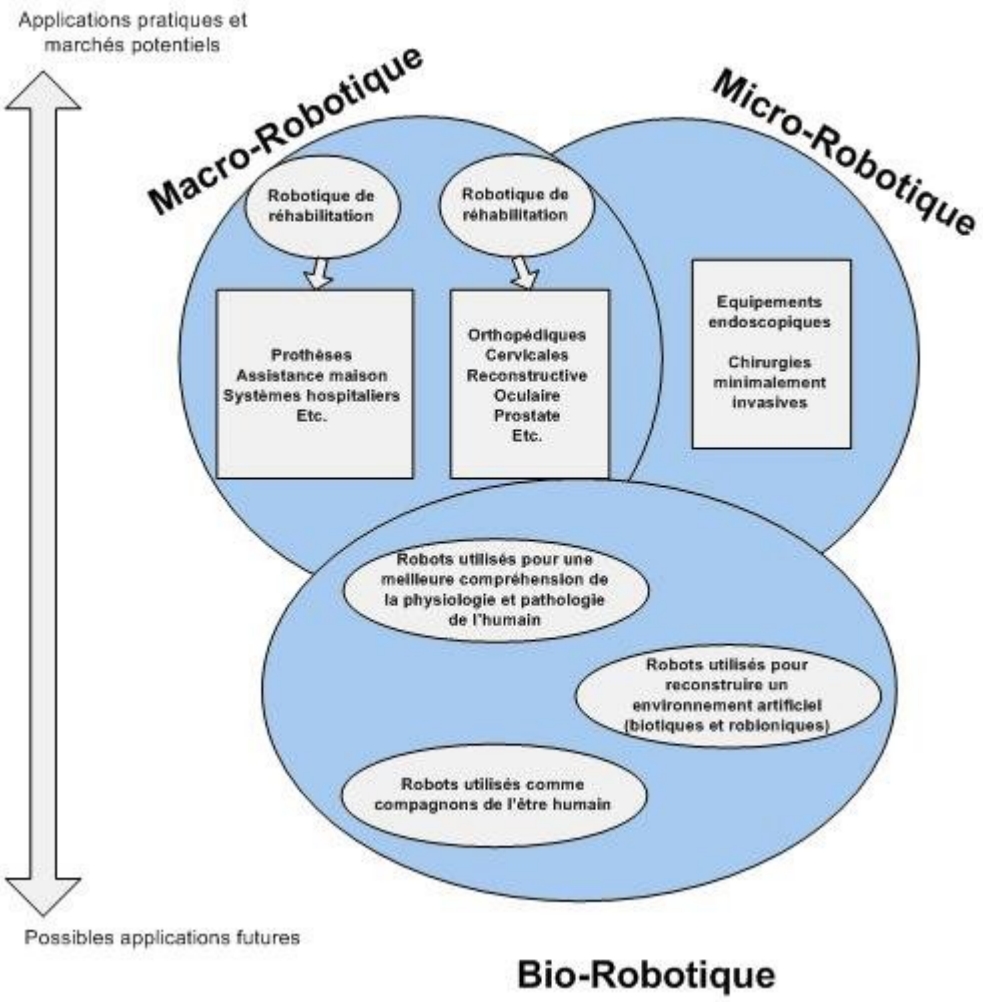


Figure 2-2. Applications et perspectives de robotique médicale **[8]**

Des exemples pratiques d'automatismes médicaux peuvent être trouvés dans des systèmes d'analyse d'électrocardiogrammes, d'imagerie à résonance magnétique, de reconnaissance de formes, de sauvegarde de données cliniques, de gestion de diagnostics, d'assistance médicale à la maison, de chirurgie assisté par robot, etc. Un des systèmes les plus connus est l'assistant automatisé en chirurgie cardiaque « DaVinci » **[15]** illustré dans la Figure 2-3. Le système est divisé en deux consoles : la première étant celle du chirurgien, où un monitorage visuel et un control précis sont offerts à travers une interface graphique, et la seconde étant celle du patient, qui consiste en deux ou trois bras robotiques contrôlant les instruments d'opération, et un bras contrôlant la vidéo endoscopique.

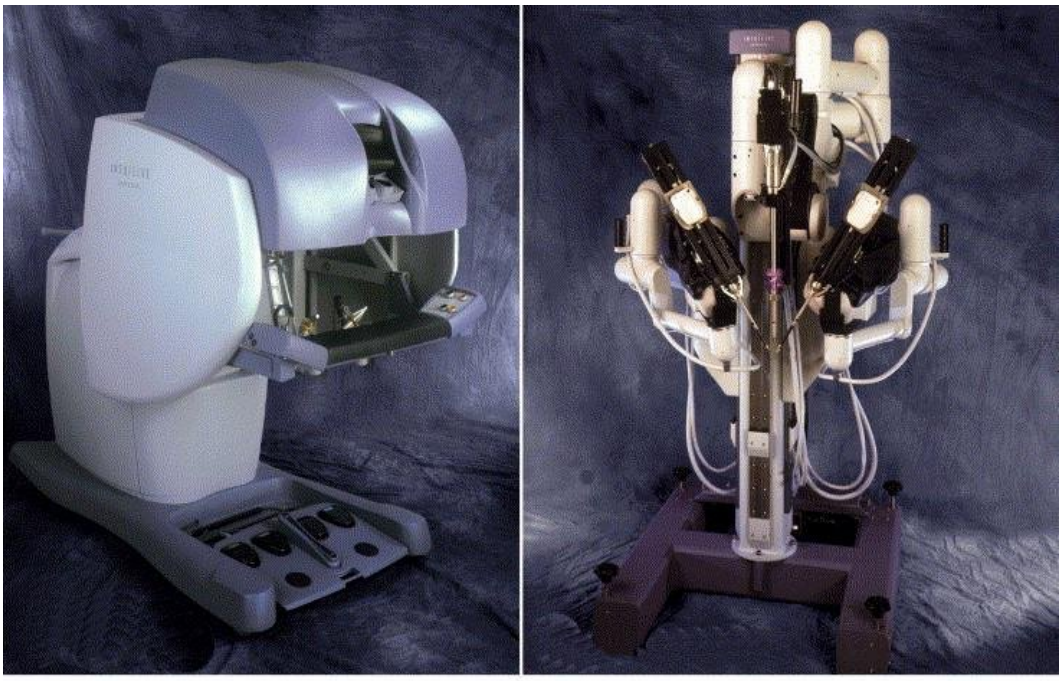


Figure 2-3. Le système de chirurgie robotique DaVinci: La console du chirurgien (A) et la console du patient (B) **[15]**

La robotique médicale offre et promet un potentiel immense pour améliorer la précision et le travail du personnel médical au cours de diverses procédures **[16]**. Cependant, c'est un domaine encore jeune et plusieurs questions de sécurité, de rendement et de prix se posent **[10]**. Des considérations majeures relatives à l'architecture des systèmes électriques, aux algorithmes de programmation utilisés, au design mécanique, à l'interface utilisateur, et à la sécurité entravent toujours les avancements rapides des systèmes automatiques dans le domaine de la médecine.

Ainsi, bien que nombreuses soient les compagnies qui fabriquent et vendent des « robots » médicaux, très peu est le nombre total de machines installées. Le marché évolue actuellement à petits pas, mais une fois les avantages bien développés et problèmes bien résolus, la médecine robotique connaîtra un essor critique et continu.

2.1.1 Automatisme vs être humain

Pourquoi alors cette évolution et utilisation d'automatismes médicaux? Qu'est-ce qui caractérise alors les systèmes automatiques et les différencie des systèmes humains?

Les systèmes automatiques ont atteint un degré de maturité assez élevé leur permettant de créer un environnement unique en salle d'opération, incluant des robots pour chirurgie locale ou télé-chirurgie, des systèmes de communication, des équipements audiovisuels interactifs, des bases de données à mise à jour automatique, etc. **[17]**. Selon **[18]**, la salle d'opération du futur sera un mélange sophistiqué de techniques d'imageries, micro robots, manipulateurs robotiques, réalité virtuelle, stations de télé-présence, médecine intégrée par ordinateur, etc.

Ainsi, la majorité des tâches médicales du présent et du futur incluront trois entités primaires : l'expert médical, le patient, et le mécanisme médical, ce dernier étant l'outil utilisé pour l'interaction, la communication, la chirurgie, l'analyse, etc. (instruments chirurgicaux, cameras endoscopiques, robot chirurgical, etc.).

Ainsi, en prenant pour base les différentes possibilités d'applications médicales, les robots peuvent être définis comme étant semi-autonomes, guidés par l'humain, ou télé-opérés (guidés à distance). Quel que soit le cas, les robots médicaux ont leurs avantages et leurs inconvénients face à l'être humain :

Tableau 2.1 Caractéristiques des humains et robots. Adapté de **[17]**

Caractéristique	Rang	Humain	Rang	Robot
Coordination	-	Limitée par les capacités du cerveau	+	Précise, basée sur des concepts géométriques
Intégration de l'information	+	Haute capacité, limitée par quantité/complexité	+	Haute capacité, limitée par algorithmes
Adaptabilité	+	Haute	-	Limitée par design

Stabilité	-	Dégradation rapide (fatigue, concentration)	+	Dégradation lente (mécanique, algorithme)
Dextérité	+	Information captée par les 5 sens	+	Haute dextérité, limitée par les senseurs attachés
Précision	-	Limitée	+	Selon besoin
Dimensions	-	Limitée au corps humain	+	Selon besoin
Échelle de travail	-	Limitée par corps humain	+	Selon besoin
Bio hasards	-	Danger de radiation, infection, etc.	+	Moins affecté par l'environnement
Spécialité	+	Générique	-	Tâches spécialisées

2.1.2 Automatisme en anesthésie

Comme dans le domaine médical, la prise de décision en anesthésie est souvent caractérisée par quelques imprécisions dues au manque d'informations ou à une mauvaise interprétation de données relatives aux patients. En effet, les paramètres usuels dépendent grandement du jugement clinique de l'anesthésiste et ne sont donc pas toujours objectifs ou standardisés **[19]**. De même qu'en médecine, les erreurs humaines sont causes de morbidité, mortalité et coûts exorbitants. Á titre d'exemple, 89% des anesthésistes de Nouvelle Zélande ont affirmé avoir fait des erreurs humaines, et 12.5% ont affirmé avoir fait souffrir un patient d'une manière ou d'une autre **[13]**. Les tâches de l'anesthésiste sont tellement nombreuses qu'il lui est quasi-impossible tout contrôler. Il doit s'occuper de tâches manuelles anesthésiques, vérifier les multiples moniteurs et paramètres, s'assurer que le patient est en bon état, collecter les informations pertinentes, gérer et analyser diverses situations qui peuvent avoir lieu, etc. (Figure 2-4)

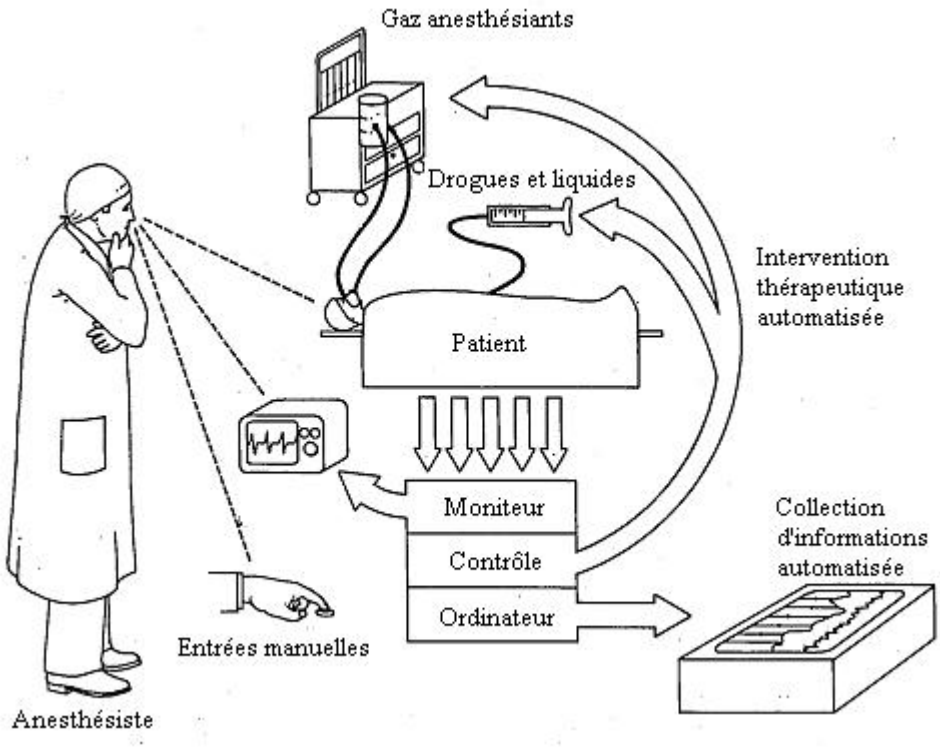


Figure 2-4. Les composantes de l'anesthésie (Adapté de **[20]**)

Certaines mesures ont été entreprises pour diminuer le taux d'erreurs humaines en anesthésie. Ainsi, des systèmes de vérification automatique de dosage ont été développés alors que d'autres systèmes enregistrent et gèrent les données vitales des patients pour une analyse en temps réel ou future **[21]**.

La Figure 2-5 met en relief un système automatique général de contrôle d'anesthésie. L'ingénieur et l'anesthésiste travaillent de pair pour étudier l'interface développée et analyser les paramètres vitaux du patient. Ces derniers sont enregistrés pour des fins de documentations, et sont parallèlement envoyées à des systèmes de contrôle en boucle fermée pour calculer ou modifier les dosages des agents anesthésiants à fournir au patient. Une fois ces dosages fournis, les nouveaux paramètres sont à nouveau vérifiés puis analysés, et ainsi de suite. Ce genre de système facilite la tâche de l'anesthésiste en lui offrant une analyse et un calcul détaillés des dosages d'agents anesthésiants à appliquer au patient **[22]**.

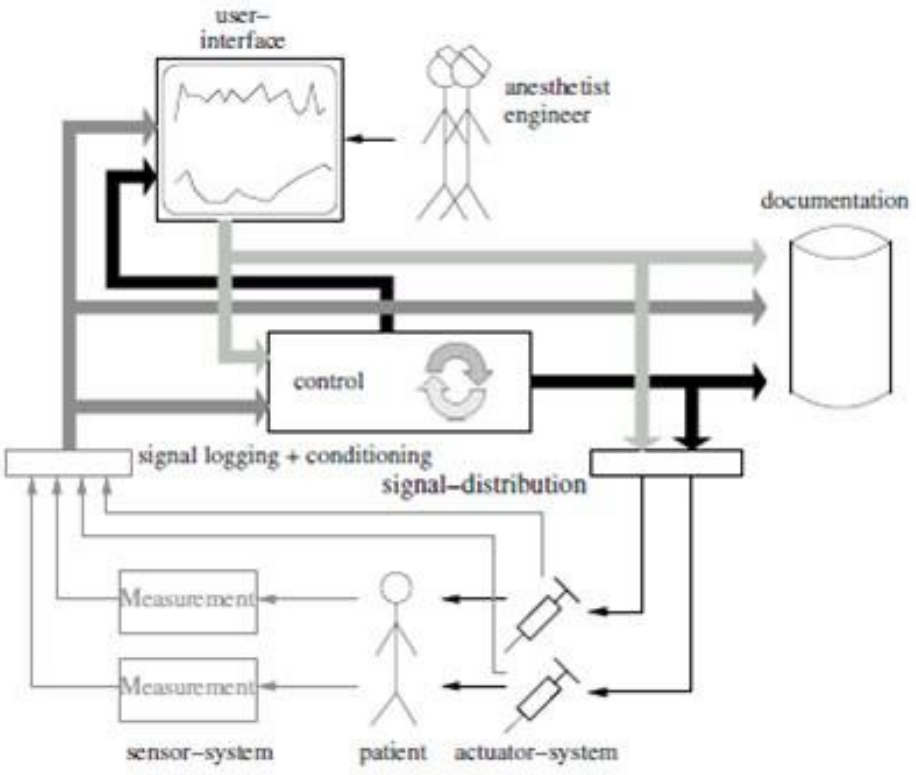


Figure 2-5. Un exemple d'automatisme général en anesthésie **[23]**

Ainsi, qu'il s'agisse de monitorage de paramètres vitaux, d'enregistrements automatiques de signaux, de base de données informatisées automatiques, de systèmes de contrôle automatique de ventilation artificielle, d'injection automatique d'agents anesthésiants, ou d'autres, les systèmes automatiques en anesthésie sont de plus en plus répandus et proposent un avenir promettant pour l'amélioration des soins aux patients et la réduction des coûts échéants.

2.1.3 Injection automatique d'agents médicaux

L'injection d'agents médicaux automatique est divisée en deux volets : mécanique et pharmacologique. Plusieurs avancements récents ont eu lieu du coté pharmacologique, mais très peu du coté mécanique.

2.1.3.1 Côté pharmacologique

Chaque agent anesthésiant est connu pour son effet sur un paramètre anatomique bien défini. Mais un effet minime peut aussi avoir lieu sur d'autres paramètres. Les drogues sont donc dites inter-fonctionnelles. La tâche de l'anesthésiste consiste donc à analyser les variations de ces différents paramètres au cours de l'anesthésie. Plus que 66% du temps de l'anesthésiste est consacré à des tâches secondaires, au lieu de se concentrer sur le bien-être du patient. Afin de faciliter le monitorage et l'analyse, des systèmes d'injection ou de contrôle automatique d'agents médicaux ont récemment vu le jour. Ces systèmes augmentent la sécurité du patient et minimisent les tâches répétitives de l'anesthésiste **[24]**. L'anesthésiste qui, en temps normal, doit surveiller plusieurs paramètres hémodynamiques et respiratoires, et signes cliniques des degrés d'hypnose, d'analgésie et de relaxation musculaire, peut alors se concentrer sur le bien-être du patient plutôt que sur le dosage des agents anesthésiants. L'inter fonctionnalité des agents anesthésiants est due à l'interaction entre multiples valeurs d'entrée et de sortie, et peut être assimilée à un problème d'ingénierie nommé « multiple input multiple output » ou « MIMO » **[23]**.

Ainsi, en se basant sur des variables relatives au patient (pression artérielle, fréquence cardiaque, etc.) et aux signaux EEG tels que le BIS (Indice bi spectral) **[25]**, des algorithmes de boucle fermée permettent l'ajustement du dosage de l'agent anesthésiant **[26]**. Aussi, en se basant sur les concentrations d'agent anesthésiant présentes dans les poumons, qui est une technique non-invasive, un ajustement peut être apporté au dosage initial **[27]**.

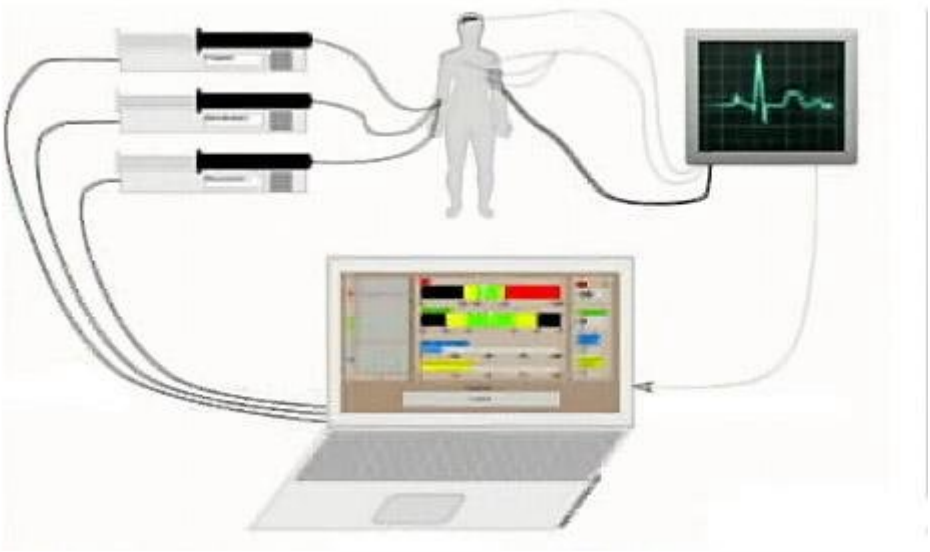


Figure 2-6. Schéma de fonctionnement de McSleepy **[28]**

La dernière percée dans le domaine, une première mondiale en Mai 2008 **[28][29]**, consiste en un algorithme informatique administrant des dosages d'agents anesthésiants de façon automatique. Ce système en boucle fermée, nommé « McSleepy », et développé au laboratoire de Dr. Thomas Hemmerling, se base sur les paramètres vitaux du patient pour calculer ou ajuster les dosages de trois agents anesthésiants contrôlant les degrés d'hypnose, d'analgésie et de relaxation musculaire. L'injection a lieu de façon automatique, libérant l'anesthésiste de tâches répétitives et lui permettant de se concentrer sur le bien-être du patient. Cependant, ce système et d'autres du genre, supposent l'aiguille ou cathéter déjà insérés dans la veine.

2.1.3.2 Côté mécanique

L'automatisation de l'aspect mécanique de l'injection d'agents médicaux ou anesthésiants est encore un domaine relativement vierge. Bien que les gestes anesthésiques sont relativement simples et peu nombreux, peu ou pas d'avancements ont été faits dans ce domaine. Ainsi, Okazawa et al ont dévéloppé un système de guidance d'aiguille pour des procédures d'invasions percutanées minimales, mais n'ont pas achevé leur système avec l'injection automatique en tant que telle **[30]**. De même, Cleary et al ont mis en place un système de positionnement précis d'un

robot guidant une aiguille pour des procédures minimalement invasives **[31]**. De plus, Hong et al ont effectué quelques tests sur un prototype d'insertion d'aiguille guidée par ultrasons **[32]**.

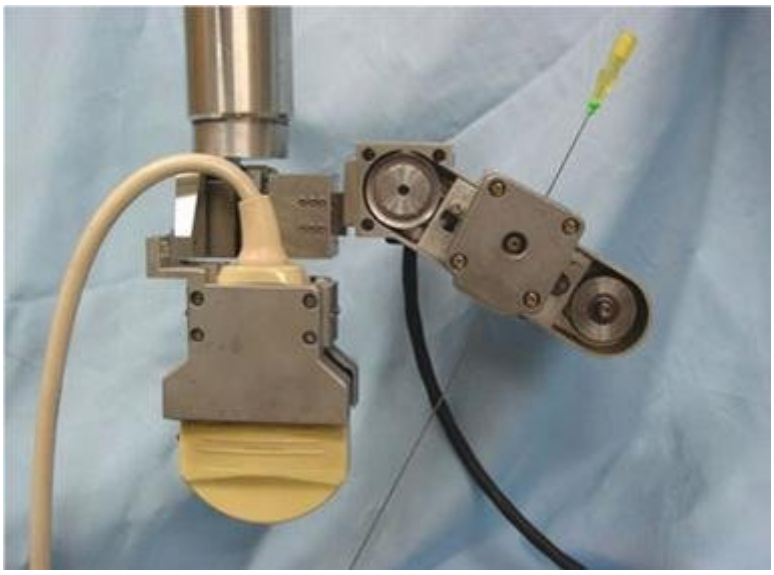


Figure 2-7. Insertion d'aiguille aidée par robot **[32]**

Dans ces trois cas, et d'autres similaires, la présence de l'être humain est nécessaire pour guider le système, et l'aiguille n'est donc pas automatiquement insérée. Le côté mécanique de l'injection automatique d'agents anesthésiants est encore un domaine qui offre nombreuses opportunités d'exploitation.

2.2 Utilisation de l'ultrasonographie en anesthésie

En cours d'anesthésie, il est nécessaire de bien visualiser l'anatomie du site d'opération afin de bien placer et doser l'agent anesthésiant. L'utilisation de l'ultrasonographie, exécutée par des mains expertes, offre une image en temps réelle de la surface à anesthésier. Ainsi, une bonne connaissance de l'anatomie humaine combinée avec une utilisation adéquate de la technique d'ultrasonographie augmente le succès de l'anesthésie. **[33]**.

Faute de ressources financières et d'habitudes opérationnelles d'anesthésistes, la transition entre les anciennes techniques et la sonographie ne se fera pas avant une dizaine d'années **[34][35]**. Depuis les premiers essais d'implantation dans le domaine de l'anesthésie, les avantages

d'utilisation de la sonographie ne font qu'augmenter cette pratique de jour en jour dans les salles d'opération. Ces avantages se résument comme suit :

- Visualisation directe des nerfs **[36][37]**.
- Visualisation des structures anatomiques (vaisseaux sanguins, muscles, os, cartilages, etc.), ce qui permet une meilleure visualisation des nerfs cachés **[38][39]**.
- Visualisation directe ou indirecte de l'expansion de l'agent anesthésiant, permettant ainsi de modifier le dosage ou l'emplacement de l'aiguille si nécessaire **[38][40]**.
- Réduction de la dose d'agent anesthésiant **[41][42]**.
- Minimisation du risque de mauvaise injection de l'agent anesthésiant **[43]**.
- Minimisation d'effets secondaires dus à une mauvaise injection de l'agent **[43]**.
- Minimisation de douleurs musculaires et réflexes dues à une mauvaise injection de l'agent **[44]**.
- Durée adéquate et meilleure qualité d'anesthésie **[42][45]**.

La technique d'ultrasonographie vise à reproduire une image de haute qualité en se basant sur des ondes sonores variant entre 5 et 14 MHz **[46]**. Les anesthésies requièrent différentes profondeurs d'images, selon l'opération ou chirurgie à entamer. Ainsi, plus la fréquence augmente, plus petite est la profondeur de l'image. Une image visualisée avec un transducteur d'ultrasons de fréquence 14 MHz est plus superficielle qu'une image visualisée avec un transducteur d'ultrasons de fréquence 8 MHz. Dans le premier cas, la peau et la surface en dessous sont apparentes, alors que dans le second cas, les veines et structures anatomiques les entourant sont mises en relief.

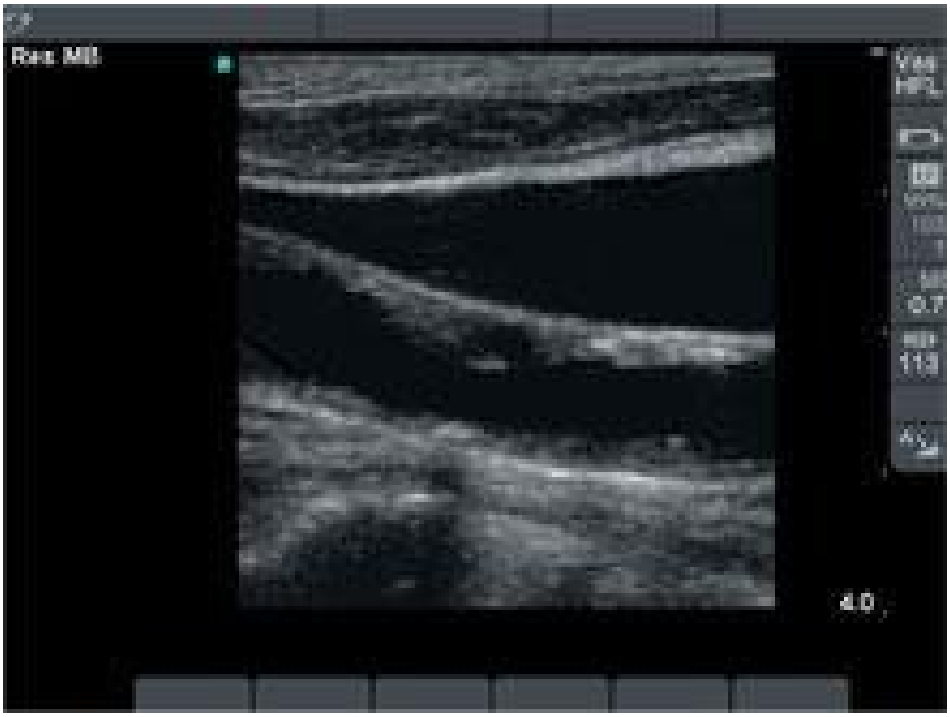


Figure 2-8. Veine jugulaire et artère carotide visualisées sur une machine d'ultrasonographie **[47]**.

De plus, le positionnement du transducteur est d'importance primordiale pour l'image à analyser. L'angle d'inclinaison et l'intensité des ondes offrent aussi des variations d'images. Enfin, des machines portables d'ultrasonographie sont présentement équipées de logiciels fournissant une haute résolution d'images, des visualisations en couleurs avec effet Doppler pour identifier les vaisseaux et flux sanguins, et une possibilité d'extraction des données soit sur CD/DVD, soit par câble (relié à un ordinateur ou analyseur), sous forme d'images ou de vidéos **[46]**.

Figure 2-9. Machine portable d'ultrasonographie. **[47]**

2.3 Matlab

Matlab est un langage de programmation de haut niveau qui offre un environnement de développement interactif permettant des computations intenses plus rapidement que d'autres langages similaires (C, C++, Fortran, etc.). Inventé par Cleve Moler vers la fin des années 70, Matlab a ensuite été développé par l'union des efforts de Cleve Moler, Jack Little et Steve Bangert en 1983. Ceux-ci ont réécrit le langage Matlab et ont mis en place la compagnie The Mathworks en 1984 pour s'occuper de la commercialisation du produit.

Le mot « Matlab » a été dérivé de MATrix LABoratory, et, comme son nom l'indique, est spécifiquement utilisé pour les tous les calculs usant de matrices et de leurs propriétés. De plus,

l'usage de Matlab est répandu pour les analyses numériques, les équations différentielles, l'analyse des signaux, l'analyse d'images, les interfaces graphiques, les modèles financiers, les tests et expériences biologiques, etc. Un des avantages principaux de Matlab est la possibilité de communiquer avec différentes applications externes (à travers les ports de l'ordinateur) ou langages externes (C, C++, Fortran, Microsoft Excel, etc.)

2.4 Transformée de Hough

La reconnaissance d'objets partiellement couverts par des irrégularités dans une image est devenue une composante importante de plusieurs problèmes de vision informatique. Habituellement, pour les cas simples, des détecteurs d'arrêtes sont utilisés pour définir des points ou lignes limitant les formes ou objets dans une image. Cependant, dans les cas complexes, certaines imperfections de l'image, ou de la détection elle-même, omettent certains points délimiteurs, donnant ainsi une mauvaise approximation de la forme à déterminer. Il serait alors plus facile d'essayer de grouper ces différentes caractéristiques extraites de la détection d'arrêtes afin de compléter la figure voulue.

La transformée Hough fut introduite pour résoudre ce problème et rendre possible ce groupement en objets « candidats » recevant chacun un score selon les similarités offertes **[48]**.

La transformée de Hough est donc une technique d'extraction de caractéristiques, utilisée généralement dans le domaine d'analyse d'images (intelligence artificielle, vision d'ordinateur, analyse d'images statiques, dynamiques, digitales, etc.) **[48]**. Le but de cette technique est d'analyser un espace de paramètres (l'image en question), afin d'y trouver des instances imparfaites concordant avec une forme géométrique (linéaire ou circulaire). Chaque groupe de pixels appartenant à l'image est examiné et une valeur correspondante est enregistrée dans une matrice nommée « accumulateur ». Les maximums locaux de l'accumulateur vont alors former l'objet recherché selon les calculs de la transformée.

La forme initiale de la transformée de Hough a été inventée par Paul Hough en 1962 pour identifier des formes linéaires dans des images **[49]**. Elle fut brevetée (U.S.Patent 3,069,654) en 1962 par IBM sous le nom « Methods and Means for Recognizing Complex Patterns ». Plus tard,

cette technique a été développée par Richard Duda et Peter Hart en 1972 pour identifier différentes formes arbitraires, notamment circulaires et elliptiques **[50]**. Duda et Hart offrent une nouvelle analyse basée sur les angles et rayons, plutôt que sur les pentes et intersections, ce qui permet des ajustements de courbes et des calculs rapides. Cette nouvelle forme fut nommée « transformée de Hough généralisée » et fut introduite au monde informatique par Dana Ballard en 1981 à travers son article « Generalizing the Hough transform to detect arbitrary shapes » **[51]**. Ballard se concentre sur la détection de formes circulaires et linéaires dans des images à niveaux de gris (« grayscale ») et prouve que la transformée de Hough permet la détection d'images complexes en les subdivisant en des arrangements de formes linéaires ou circulaires.

Un des problèmes les plus récurrents en analyse d'images informatiques est la détection de lignes droites. Dans les plus simples cas, la ligne est formée d'une succession de points noirs distincts sur un fond blanc. Ce problème est généralement résolu en comparant les droites respectives aux différentes combinaisons de paires de points **[50]**. Ainsi, dans un espace paramétrique, Hough utilise l'équation de droite $y = mx + b$ pour dessiner graphiquement chaque paire de points (x,y) appartenant à l'espace défini.

Cependant, et pour simplifier les calculs, Hough traduit chaque équation de droite par deux paramètres : la distance à l'origine « r » et l'angle de la distance à l'origine « θ »:

$y = \left(-\frac{\cos\theta}{\sin\theta} \right) x + \left(\frac{r}{\sin\theta} \right)$ qui peut être réécrite $r = x\cos\theta + y\sin\theta$ **[48]**. Ainsi, à chaque droite trouvée, un couple unique (r, θ) peut être associé, quand les conditions de base sont respectées ($\theta \in [0, \pi]$ et $r \in R$ ou $\theta \in [0, 2\pi]$ et $r \geq 0$). De plus, sachant qu'à travers chaque point (x0, y0) du plan peuvent passer une infinité de droites, ces droites doivent toutes satisfaire l'équation : $r(\theta) = x_0 . \cos\theta + y_0 . \sin\theta$, qui n'est autre qu'une sinusoïde unique à ce point dans le plan (r, θ). Ainsi, le problème de trouver des points colinéaires se transforme en un problème de trouver des courbes sinusoïdales concurrentes. **[52]**. De cette façon, à chaque point possible de l'espace, la transformée de Hough évalue la concordance des courbes sinusoïdales, et les points retenus sont enregistrés dans une matrice nommée « accumulateur ».

2.5 Interface utilisateur

2.5.1 Concepts de base d'une interface graphique

Les interfaces graphiques, telles que présentement connues, influencent grandement le travail et communication des personnes usant de supports informatiques. Elles sont devenues comme une couche opaque, cachant la complexité des algorithmes, pour rendre accessibles diverses fonctionnalités sans que l'utilisateur n'ait à comprendre l'essence des codes cachés. L'évolution des interfaces graphiques a vu non seulement plusieurs histoires à succès (Xerox STAR, Apple LISA, Microsoft Windows, Apple Mac OS, etc.), mais aussi plusieurs échecs (Microsoft Windows 1.0, Microsoft BOB, Tandy Deskmate, etc.). Faisant face à un marché de matériel informatique (« hardware ») en développement continu, les interfaces graphiques n'ont guère évolué au cours des dernières quelques années. Les concepts de base des interfaces sont quasi-inchangés et ce ne sont que les options, ou fonctionnalités supplémentaires qui diffèrent d'une interface à une autre. A travers ces différents succès et échecs, des standards de base pour interfaces graphiques sont maintenant répandus **[53]**. Ces standards sont basés sur des concepts de psychologie cognitive et ont été prouvés avec des années de tests et d'expérience :

- Consistance : Une interface graphique doit être consistante avec le contenu qu'elle reflète. Les applications et options doivent évoquer les mêmes fonctions et résultats et offrir une consistance externe et interne pour une communication saine entre l'être humain et l'interface graphique.
- Métaphores : Afin d'aider à la compréhension de l'utilisateur, des métaphores sont utilisées pour banaliser des complexités (algorithmes techniques) et les rendre assimilables. Ex. : sur une interface Windows, on appelle « bureau » l'endroit où les fichiers principaux (« mon ordinateur », « poubelle », etc.) sont disposés. L'utilisation de sons, images, boutons et actions peut renforcer ces métaphores et faciliter l'usage de l'interface.
- Centré sur l'utilisateur : Bien qu'il ne soit pas nécessairement expert en programmation, celui-ci doit toujours se sentir responsable de l'interaction et des différentes étapes de

l'interface. Ainsi, les actions doivent être exécutées et contrôlées par l'utilisateur, à travers des boutons « métaphoriques ». Les résultats visibles confortent l'utilisateur et confirment ses actions. Toute erreur doit être signalée et l'interface doit offrir un choix, ou alternative, pour corriger ou détourner l'erreur.

- Esthétique : L'esthétique de l'interface est non-négligeable. L'être humain peut facilement être désorienté par une interface mal organisée. L'utilisation de boutons, de tableaux, de hiérarchie, de couleurs, le groupement d'objets similaires, etc. ont un effet important sur la capacité de compréhension de l'utilisateur et son utilisation de l'interface, et donc du produit.

2.5.2 Interfaces graphiques en anesthésie

En anesthésie, la surcharge d'information due au monitorage des patients peut être comparée à celle de l'aviation. Il faut donc optimiser les moniteurs de l'état du patient pour faciliter la prise de décision des anesthésistes. Le but d'une interface est donc de bien mettre en évidence les aspects importants de l'anesthésie afin d'éviter les erreurs d'interprétation et faciliter la prise de décision. En effet, les erreurs associées au mauvais jugement de l'anesthésiste forment 80% des accidents critiques **[54][55][56]**. Une interface graphique basée sur des objets graphiques, plutôt que sur du texte offre un meilleur rendement en terme de performance et d'erreurs **[57][58]**. Une des études les mieux détaillées est celle de Gurushanthaiah et al. **[59]**, où les auteurs comparent trois formats d'affichage pour le monitorage de paramètres physiologiques en anesthésie : numérique, histogramme et polygonal. Dans le cas de l'affichage numérique, les valeurs mesurées sont disposées numériquement seulement. L'anesthésiste identifie la variable voulue en un temps relativement normal. Le second cas offre un histogramme (diagramme à colonnes ou à barres) en plus du format numérique, ce qui donne une représentation plus ou moins graphique des variables étudiées. Le troisième cas, quant à lui, consiste en une figure géométrique polygonale, où chaque sommet représente une valeur. La forme de la figure géométrique varie proportionnellement au changement des variables mesurées. L'étude de Gurushanthaiah montre que les formats « histogramme » et « polygone » aident l'anesthésiste à une meilleure identification des variables, à une meilleure analyse des variations de ces variables, et à une précision de réponses supérieure à celle résultant de l'affichage numérique. Suite à ces résultats,

deux études importantes vinrent confirmer l'importance de l'utilisation d'affichages graphiques en plus de l'affichage numérique. Ainsi, Michels et al **[60]** ont comparé un affichage graphique de plusieurs paramètres avec l'affichage numérique, en simulant quatre évènements critiques (perte sanguine, paralysie inadéquate due à la ventilation, fuite du ballonnet et épuisement de la chaux sodée). Blike et al **[61]** ont simulé un contexte opératoire avec une interface graphique, dans le but de réduire la surcharge d'information, minimiser la demande temporelle mentale et physique, et augmenter la précision des diagnostics et la détection d'évènements critiques chez l'anesthésiste. Dans les deux cas, deux groupes d'anesthésistes ont été testés. Ces derniers ont détecté les évènements importants plus rapidement en observant l'affichage graphique qu'en observant l'affichage numérique.

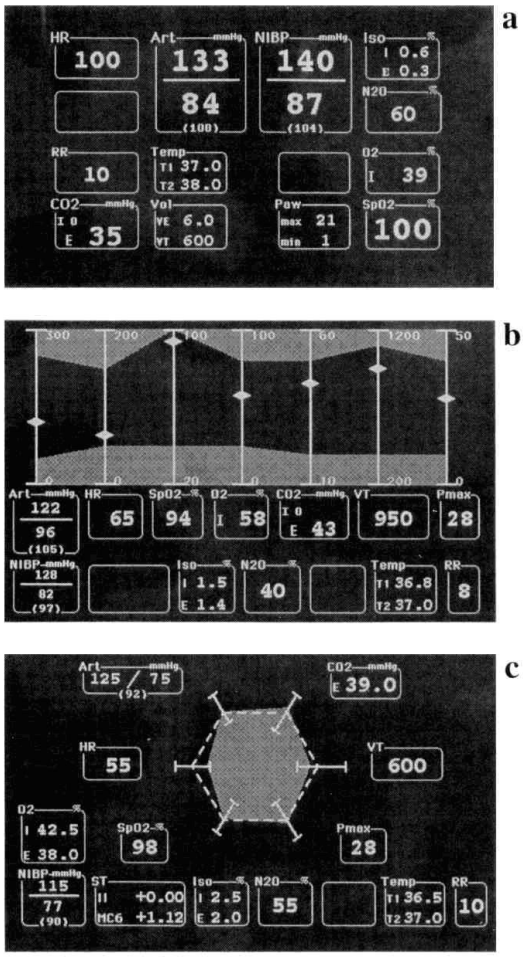


Figure 2-10. L'affichage en mode numérique (A), histogramme (B) et polygonal (C) utilisé par le stimulateur clinique. L'identification de la variation d'une variable et le sens de cette variation ont été détectés plus rapidement lors de l'affichage de l'histogramme ou du polygone **[59]**

CHAPITRE 3 MÉTHODOLOGIE

3.1 Algorithme

L'algorithme suivant illustre le schéma expérimental résumant le projet en gros.

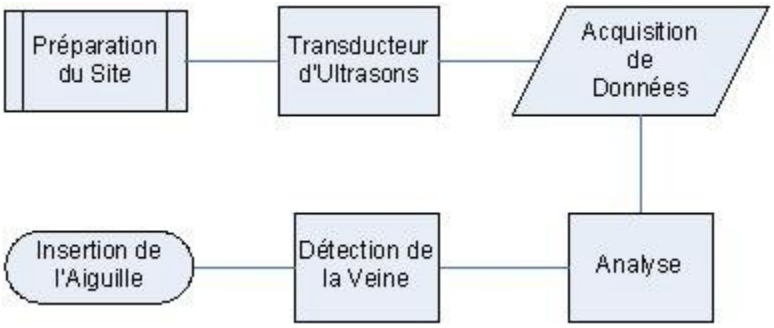


Figure 3-1. Algorithme général du projet

Détection de la veine par Ultrasons : La main du patient est premièrement immobilisée. Elle est ensuite préparée pour un bon rendement d'images ultrasonographiques : un bon nettoyage et une généreuse application de gel conducteur sont essentiels pour une image claire. Par suite, une vidéo d'images d'ultrasons est projetée sur écran et des coupes sont prélevées. L'analyse de ces coupes (images distinctes) se fait en temps réel. Chaque image est traitée et la veine est visuellement localisée.

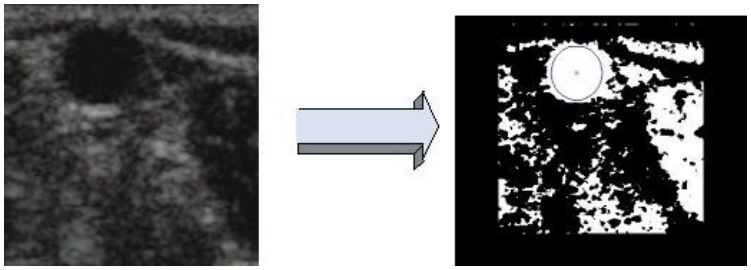


Figure 3-2. Traitement et analyse d'image. *A gauche*: Image ultrasonographique; *à droite*: veine détectée

Traitement et Analyse d'Images : Les images obtenues par ultrasons sont alors directement transférées à un algorithme de traitement et analyse d'images, développé sur Matlab. Elles sont échantillonnées et un traitement a lieu pour déterminer l'emplacement exact, et donc les coordonnées, de la veine sur chaque image. Une analyse tridimensionnelle est effectuée afin de s'assurer de la validité de la veine, en comparant les coordonnées relatives aux images traitées. Il est alors possible de détecter la présence d'embranchements ou non, la courbure de la veine, etc. pour pouvoir valider l'insertion subséquente.

Insertion de l'Aiguille : Une fois les coordonnées de la veine déterminées, elles sont traduites et envoyées à un système automatique (formé de microcontrôleur et moteurs) qui les traduit en un mouvement précis de la seringue pour la positionner et la guider. La peau, puis la veine, sont ainsi transpercées, pour placer le cathéter et administrer l'anesthésiant.

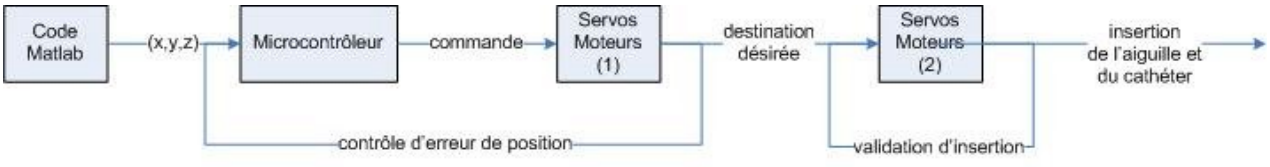


Figure 3-3. Des coordonnées de la veine, à l'insertion de l'aiguille

Les illustrations ci-dessous exposent les dessins schématiques du dispositif expérimental du projet :

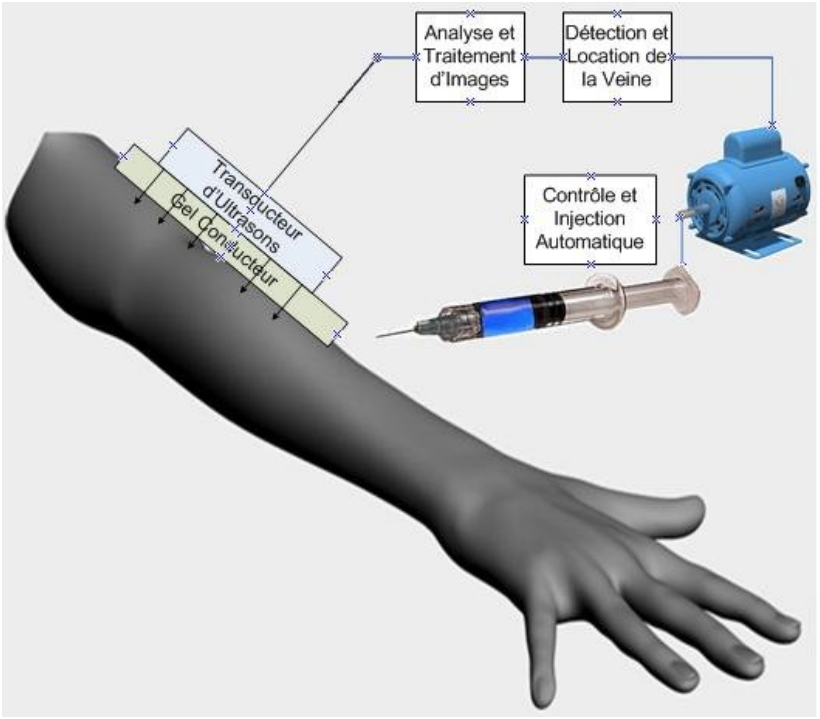


Figure 3-4. Dispositif Expérimental

Il est primordial de noter qu'un modèle de bras est utilisé pour ce projet. La veine cubitale est simulée par un tube rempli d'eau, transperçant une forme cylindrique gélatineuse (bras).

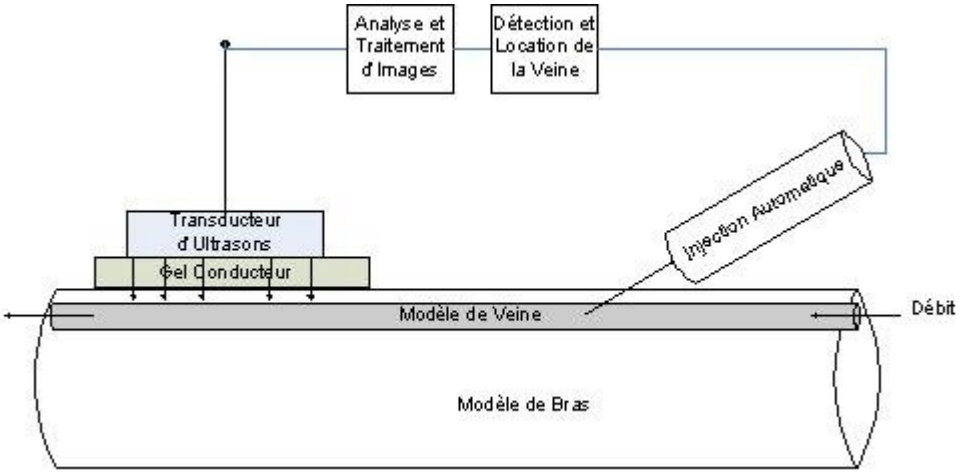


Figure 3-5. Modèle de Simulation du Bras et de la Veine

Les sections suivantes décriront en plus de détails la méthodologie utilisée pour les différentes composantes du projet.

3.2 Premier bras robotique

Pour assurer un mouvement automatique, il est nécessaire de construire un système de bras robotiques manipulés par les lois de cinématique. Ce système, autonome en grande partie (l'anesthésiste devra quand même introduire des données et s'assurer de la sécurité et du fonctionnement correct), est mis en place pour différentes parties du corps humain et diverses applications dans le domaine de l'anesthésie. De plus, il devra être capable d'opérer en dehors des salles d'opérations, et par un personnel initié.

Ainsi, un premier bras robotique devra assurer le mouvement automatique de la sonde d'ultrasons (L25, Sonosite, Bothell, WA, États-Unis), qui survolera le bras du patient afin de contrôler les différentes prises de coupes successives sur le site à opérer. Le site d'étude (pour ce projet, le site de tests consiste en un bras fantôme comportant une veine fantôme) devra premièrement être nettoyé puis rendu conducteur avec l'application d'un gel médical conducteur qui va garantir une bonne image ultrasons. La sonde prendra plusieurs images à 1.5cm d'intervalle afin d'offrir une meilleure étude de la veine en question, images qui seront observées sur l'écran d'une machine portable d'ultrasons, Sonosite Micromaxx (Sonosite, Bothell, WA, États-Unis).

Pour ce, le bras robotique doit assurer un positionnement facile de la sonde ultrasons dans l'espace. Six moteurs Hitec (Hitec, RCD, Poway, CA, États-Unis) s'occuperont d'un positionnement initial de la sonde dans l'espace selon l'illustration suivante :

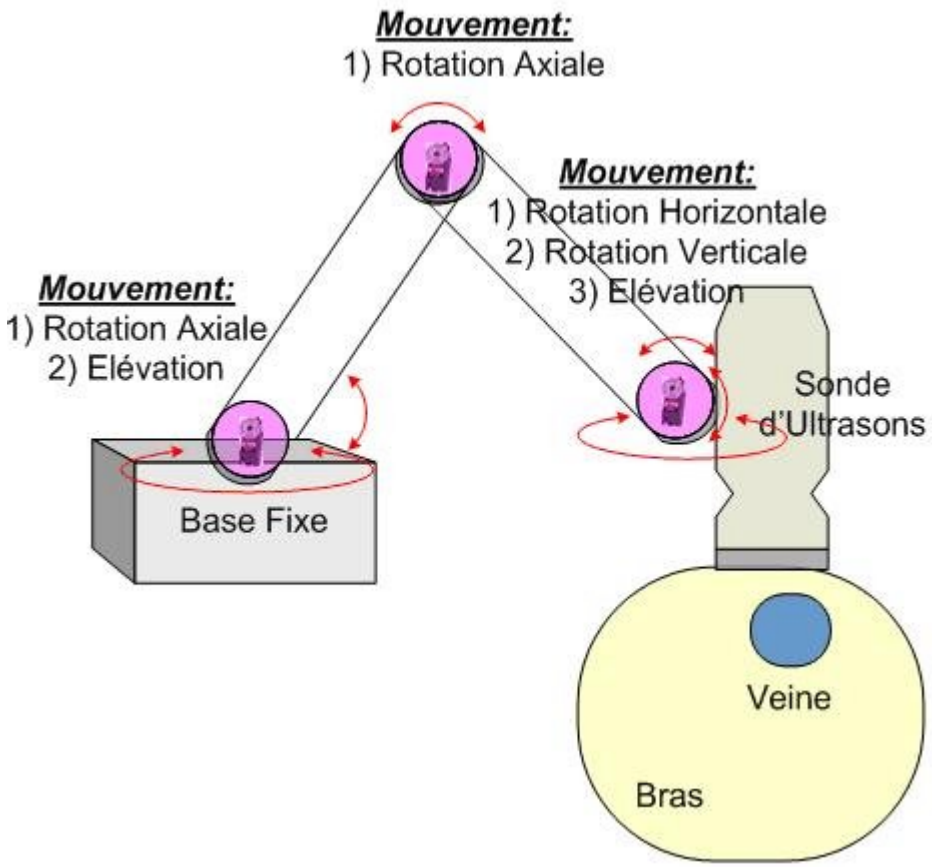


Figure 3-6. Schéma du premier bras robotique: 6 moteurs assurent les différents mouvements nécessaires au positionnement de la sonde d'ultrasons

Une coordination des mouvements des moteurs 1 à 6 aide au positionnement désiré de la sonde ultrasons et à son mouvement vers les différents emplacements subséquents. Ainsi la sonde pourra se déplacer et différentes coupes peuvent alors être enregistrées pour une future analyse.

3.3 Équipements de Communication

Comme déjà précisé, la sonde ultrasons « L25 » de « Sonosite » projette les images sur l'écran de la machine portable d'ultrasons. Ces images dynamiques sont transmises à un convertisseur vidéo multifonctionnel de haute qualité, un « Canopus TwinPact100 » (Canopus, San Jose, CA, États-Unis), à travers un câble « S-Video ».

Figure 3-7. Acquisition des données

Un câble « firewire » transmet alors les images capturées, qui sont maintenant lues en données vidéo, à un ordinateur où le programme « Matlab » (The Mathworks, Natick, MA, Etats-Unis) les visualise sur l'écran de l'ordinateur pour y montrer les images ultrasons et les coupes automatiques en temps réel.

La communication entre l'ordinateur et les moteurs de contrôle se fait à travers un contrôleur « servo » SSC-32 de la compagnie Lynxmotion. Avec une précision de résolution de 1µs, ce contrôleur servo permet une communication bidirectionnelle avec jusqu'à 32 moteurs en mouvement synchrone ou de groupe.

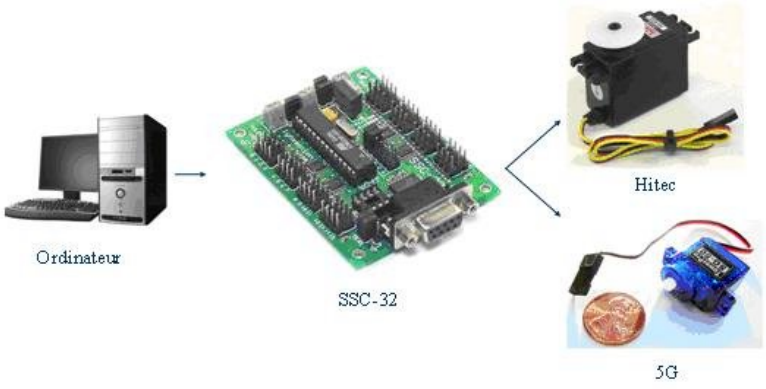


Figure 3-8. Transfert des données

3.4 Algorithme d'analyse Matlab

Matlab étant un outil d'analyse solide et permettant des calculs assez rapides, il a été choisi parmi d'autres pour traiter les multiples analyses et mener le projet à terme. L'algorithme fonctionne comme suit :

3.4.1 Connexion, balayage et transfert de données

1. Matlab se connecte au port série

```
Ser=serial('COM1', 'BaudRate',115200,'Terminator', 'CR/LF');
                        .
                        .
```

2. Les commandes d'initialisation sont envoyées au premier bras robotique.

```
fprintf(Ser,MALLini);
```

3. MALLini est un regroupement de plusieurs commandes chacune contrôlant le mouvement d'un des moteurs Hitec composant le premier bras robotique. L'agencement de ces commandes permet de coordonner le positionnement initial du bras et de se refermer autour de la sonde ultrasonographique.

4. Celui-ci balaye le site d'injection (le bras fantôme) avec la sonde d'ultrasonographie. Les images sont projetées en temps réel sur la machine portable d'ultrasonographie, Micromaxx de Sonosite.

5. Similaire à MALLini, MALLarm1 est un regroupement de plusieurs commandes des moteurs Hitec du premier bras robotique afin de coordonner le balayage de ce dernier au-dessus du site d'injection.

```
fprintf(Ser,MALLarm1);
```

6. Les données vidéo sont transmises au programme Matlab pour analyse.

```
obj = videoinput('winvideo', 1);

src_obj = getselectedsource(obj);
```

7. Une interface graphique a été élaborée pour permettre à l'utilisateur de suivre l'analyse et en valider les étapes. Cette interface est illustrée dans la prochaine section.

3.4.2 Analyse d'images et sélection de coordonnées

8. La vidéo obtenue est visualisée et des coupes (images fixes), à intervalles prédéfinis par les mouvements du premier bras robotique, sont enregistrées.

```
S = [S; getsnapshot(obj)];
```

9. Il est à noter qu'il est possible de lancer des images préenregistrées en mémoire d'ordinateur, ou à partir de médiums d'enregistrement (CD, DVD, Clé USB, etc.). Dans ce cas là, l'analyse est passive et ne peut être considérée en temps réel.

```
S = [S; imread([pathname filename])];
```

10. Avant de procéder à l'analyse subséquente de l'image, celle-ci doit être coupée pour en éliminer les bordures.

```
S{imgcount} = imcrop(S{imgcount},[UL ul LR lr]);
```

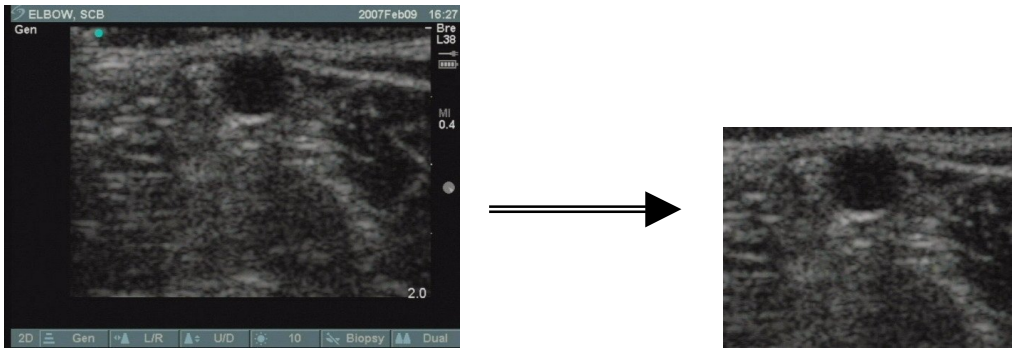


Figure 3-9. Exemple d'image avec et sans bordure

11. L'image doit ensuite être démunie de couleurs et convertie en noir et blanc pour rendre l'analyse facile, rapide, et supportée par la mémoire vive de l'ordinateur et de Matlab.

```
Y = im2bw(256-I,0.75);
```

12. Puis, l'image est filtrée (medfilt2) sur ses deux dimensions et le bruit est traité selon la dimension linéaire verticale.

```
Y=medfilt2(Y);

se = strel('line',11,90);

I = imerode(Y,se);
```

13. Chaque image fixe est analysée pour détecter l'emplacement de la veine sur cette image; la veine apparaît comme une forme circulaire sur l'image.

```
        HM = zeros(sy*sx,1);
.
        a = zeros(sy,totalpix);
.
        ind1 = sub2ind([sy,sx],b1,a1);
        ind2 = sub2ind([sy,sx],b2,a2);

        ind = [ind1; ind2];
.
        val = ones(length(ind),1);
        data=accumarray(ind,val);
        HM(1:length(data)) = data;
        HM2 = reshape(HM,[sy,sx]);
.
        [maxval, maxind] = max(max(HM2));
        [B,A] = find(HM2==maxval);
.
```

14. A chaque emplacement détecté sont assignées des coordonnées relatives à l'image en question, coordonnées qui seront jumelées à l'emplacement plus général de la sonde ultrasons pour finalement détecter le positionnement exact à atteindre.

```
        mean(A), mean(B);
.
.
            cylx=[cylx, mean(A)];
```

15. Ainsi, les images sont premièrement enregistrées dans une matrice, chaque image étant un singleton appartenant à cette matrice, puis converties en noir et blanc selon une valeur limite de 0.75. Cette conversion facilite la prochaine étape, la transformée de Hough. En effet, appliquer la transformée de Hough sur une image couleur consume la mémoire vive de l'ordinateur et prend un temps important, la mémoire étant même insuffisante dans certains cas, et fournit des résultats erronés. Par contre, une transformée de Hough appliquée à la même image en noir et blanc peut prendre quelques secondes, selon la valeur limite prédéfinie. Cette étude permet de trouver la forme circulaire dans l'image (la veine) et par suite en déterminer les coordonnées par rapport à l'image en question (et par conséquent, par rapport a l'espace 3-D).

```
          if(deltacenters>thresh) %if vein varies a lot at some
point


              successful=0;
              break;


          else
              successful=1;
```

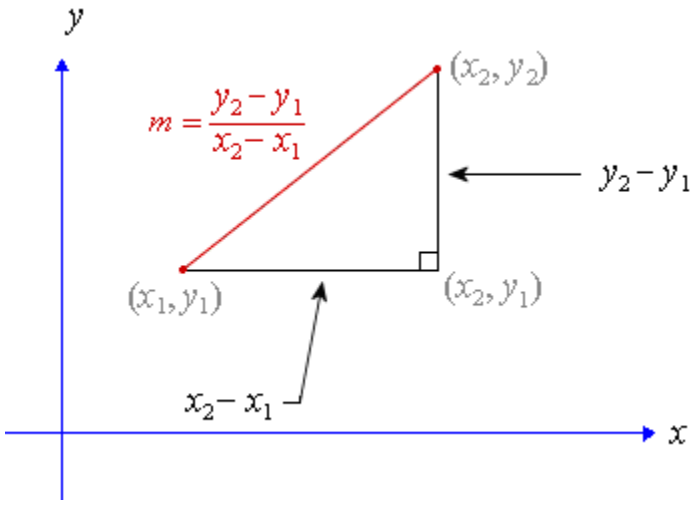
3.4.3 Détection des angles du second bras et insertion de l'aiguille

L'analyse d'images ultrasons permet de trouver les coordonnées des centres de veine sur les différentes coupes prises. Afin de trouver les angles d'insertion de l'aiguille dans le plan horizontal et vertical, nous devons déterminer les pentes respectives dans ces deux plans, relativement aux points représentant les centres de veines.

Par définition, dans un plan xy, la pente est définie par la formule :

$$m = \frac{y_2 - y_1}{x_2 - x_1};$$

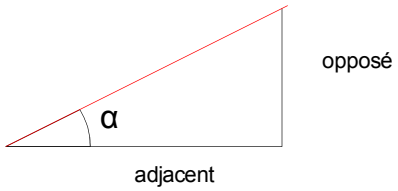
où (x1, y1) et (x2, y2) sont les coordonnées respectives des points entre lesquels nous déterminons la pente. Le schéma ci-dessous illustre la pente calculée entre deux points.



En d'autres termes, la pente étudie le rapport entre la variation verticale (Δy) et la variation horizontale (Δx).

Nous savons aussi que la tangente d'un angle est le rapport du coté opposé sur le coté adjacent. D'où la formule :

$$m = \tan\alpha = \frac{opposé}{adjacent}$$

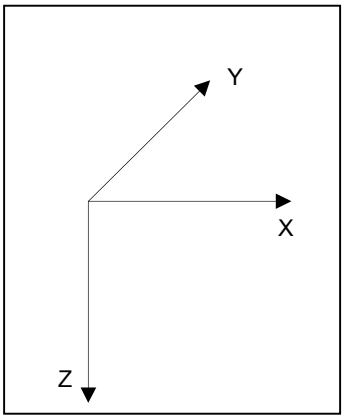


Par conséquent nous pouvons trouver la valeur de l'angle α à travers l'inverse de la tangente, ou l'arc tangente :

$$\alpha = \tan^{-1} m = \arctan m$$

Afin de déterminer les angles d'insertion de l'aiguille dans les deux plans horizontal et vertical, ce raisonnement doit être appliqué à ces deux plans respectifs.

Le référentiel utilisé est illustré dans la figure ci-contre, et a pour centre le point O, base du bras robotique : Y est aligné avec le bras du patient, X est la distance du point de base du bras robotique et perpendiculairement au bras du patient, et Z est la profondeur de la veine (à la surface de la peau).



L'angle vertical est donc déterminé par le rapport des variations d'éloignement (Δy) et de profondeur (Δz). Ainsi :

$$m_{verticale} = \frac{y_2 - y_1}{z_2 - z_1} \text{ et } \alpha_{vertical} = \tan^{-1} m_{verticale} = \arctan m_{verticale}$$

L'angle horizontal est donc déterminé par le rapport des variations d'éloignement (Δy) et de profondeur (Δz). Ainsi :

$$m_{horizontale} = \frac{y_2 - y_1}{z_2 - z_1} \text{ et } \alpha_{horizontal} = \tan^{-1} m_{horizontale} = \arctan m_{horizontale}$$

En se basant sur ces principes, il est possible alors d'écrire :

```
slopeH=(cyly(2)-cyly(1))/((cylx(2)-cylx(1))/215);
alphaH=(180/pi)*atan(slopeH);
.
.
slopeV=((cylz(2)-cylz(1))/200)/(cyly(2)-cyly(1));
alphaV=(180/pi)*atan(slopeV);
```

L'analyse des différentes prises de coupes permet de déterminer les pentes et angles horizontaux et verticaux de la veine. Ainsi, en utilisant les lois de cinématique inverse, le positionnement des différents moteurs et tiges métalliques du second bras robotique est calculé pour placer l'aiguille et le cathéter conformément au-dessus de la veine détectée.

```
if alphaH < 0
    Xb=Xa+(-0.5*NL*cos(alphaV*pi/180)*sin((90+alphaH)*pi/180));
    Yb=Ya+(-NL*cos(alphaV*pi/180)*cos((90+alphaH)*pi/180));

else
    Xb=Xa+(+0.5*NL*cos(alphaV*pi/180)*sin((90-alphaH)*pi/180));
    Yb=Ya+(-NL*cos(alphaV*pi/180)*cos((90-alphaH)*pi/180));
end

Zb=Za+(NL*sin(alphaV*pi/180));
```

Une fois les coordonnées de la veine déterminées, elles seront envoyées à un système automatique qui s'occupera de les traduire en un mouvement précis de l'aiguille de la seringue pour la positionner et la guider à transpercer la peau puis la veine et administrer l'anesthésiant.

```
d=sqrt((Xb^2)+(Yb^2)+(Zb^2));


betaM8=(180*asin(Zb/d))/pi; % Elevation angle for Motor M8


angleyOB=(180*acos(Xb/d))/pi;


angleM7=(180*acos((d^2 + s1^2 - s2^2)/(2*d*s1)))/pi; % Base
rotation angle
anglexOB=(180*asin(Yb/d))/pi;
betaM7=90-(anglexOB + angleM7);


betaM9=(180*acos((-d^2 + s1^2 + s2^2)/(2*s2*s1)))/pi; %ODB
angle
```

```
PbetaM7=1600-(angleM7/0.12);

SbetaM7=int2str(PbetaM7);

PosbetaM7=strcat('#7P',SbetaM7,'T10000');


PbetaM8=1970-(betaM8/0.12);

SbetaM8=int2str(PbetaM8);

PosbetaM8=strcat('#8P',SbetaM8,'T10000');


PbetaM9=2150-((180-betaM9)/0.12);

SbetaM9=int2str(PbetaM9);

PosbetaM9=strcat('#9P',SbetaM9,'T10000');
```

3.5 Interface Graphique

Afin de permettre à l'utilisateur de suivre les différentes étapes de cette analyse Matlab et d'en valider les résultats successifs, une interface graphique simple doit être mise en place. Cette interface doit contenir les différentes composantes du projet et être capable de les mettre en relief d'une façon assez facile et pratique. L'interface élaborée a été évaluée et approuvée par deux anesthésistes et trois ingénieurs travaillant dans le même environnement.

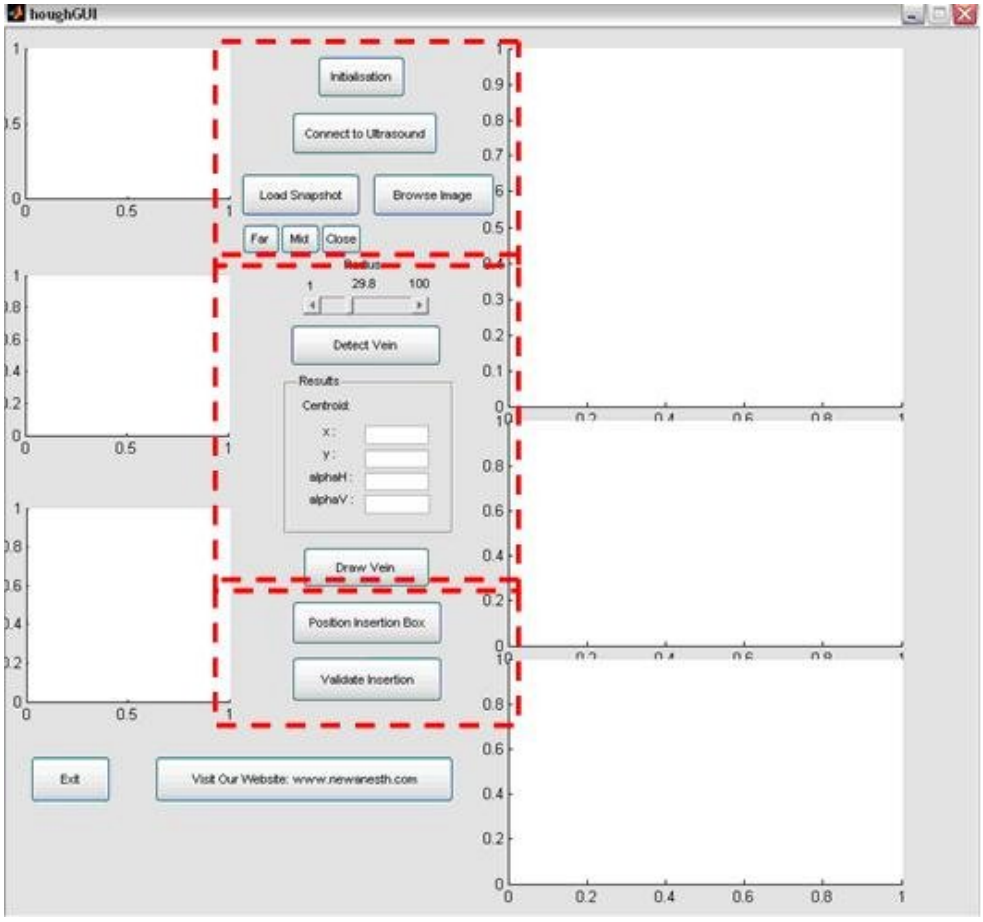


Figure 3-10. Division de l'interface en trois parties

Une division tripartite fut donc entreprise :

- Une partie initiale initialise le premier bras robotique pour qu'il se positionne, se connecte à la machine portable d'ultrasons Sonosite MicroMaxx, et enregistre les images sélectionnées (temps réel ou temps passif).
- La seconde partie s'occupe de l'analyse d'images en les convertissant en noir et blanc puis en appliquant la transformée de Hough et ressortir les coordonnées et angles d'inclinaisons de la veine, relativement au positionnement de la sonde.
- Enfin, la troisième partie prend en charge la coordination des mouvements du second bras robotique, en se basant sur les lois de cinématique inverse, pour positionner le bras au point d'insertion et insérer l'aiguille, puis le cathéter dans la veine.

3.5.1 Initialisation et prise d'images

Ainsi, à travers l'interface graphique, l'utilisateur devra premièrement initialiser les mouvements du premier bras robotique pour qu'il se positionne correctement au dessus du bras du patient et performe le balayage de la probe d'ultrasons pour collecter les données vidéo. En appuyant sur l'onglet « Initialisation », les informations préenregistrées sont envoyées à la carte SSC-32, à travers un câble RS-232, et transmises aux moteurs gouvernant le premier bras robotique. Ce dernier se place en position initiale au dessus du site d'importance, généralement l'avant-bras.

La pince à l'extrémité du bras robotique attend que l'utilisateur appuie sur l'onglet « Connect to Ultrasound » (connexion à la machine d'ultrasons) pour se refermer sur la sonde et la fixer en place. Simultanément, une connexion est établie entre Matlab et la Sonosite Micromaxx, et une fenêtre vidéo de Matlab apparait pour reproduire, en temps réel, la suite d'images ultrasons visualisée sur l'écran Micromaxx.

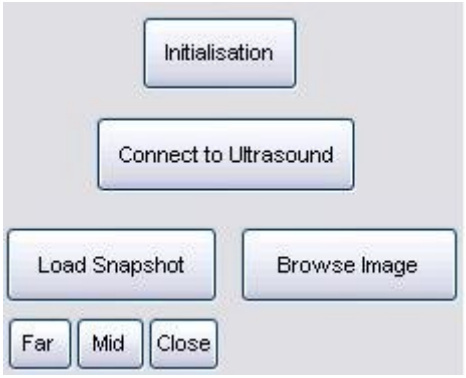


Figure 3-11. Initialisation et capture d'images

Trois choix s'offrent alors à l'utilisateur :

- « Load Snapshot » permet de charger la matrice d'images fixes prises directement de la vidéo, c'est-à-dire en temps réel. De l'emplacement initial, une première image est enregistrée, puis le bras robotique effectue un mouvement souple pour glisser la sonde ultrasons de quelques millimètres et prendre automatiquement une seconde image.
- « Far », « Mid » et « Close » performent les mêmes étapes de chargement d'images que « Load Snapshot » mais à des distances prédéfinies, mais différentes. Ainsi, « Far » positionne la sonde ultrasons à y = 16.5 cm, « Mid » à y = 15 cm, et « Close » à y = 13.5

cm, les valeurs des coordonnées x et z étant les mêmes puisqu'on suppose que le bras fantôme ne change pas de position.

- « Browse Image » permet de charger des images préenregistrées sur l'ordinateur ou sur n'importe quel autre médium de stockage relié à l'ordinateur (CD-ROM, DVD, Clé USB, diskette, disque dur externe, etc.). Il est à noter que cette option peut être jumelée aux deux options précédentes pour analyser simultanément des images en temps réel et d'autres en temps passif.

3.5.2 Détection de la veine

Une fois les images stockées dans la mémoire vive, l'utilisateur choisit le rayon de la veine, en pixels. Selon la qualité de l'image, le rayon diffère et peut avoir diverses valeurs pour une même image visualisée à des profondeurs de champs divergentes. Pour ce projet, étant donné que la sonde ultrasons est la même, que le bras fantôme est immobile, et que la veine fantôme utilisée est inchangeable, la résolution de l'image résultante ne diverge pas et le rayon a une valeur proche de 30 pixels. L'analyse est déclenchée en cliquant sur « Detect Vein ».

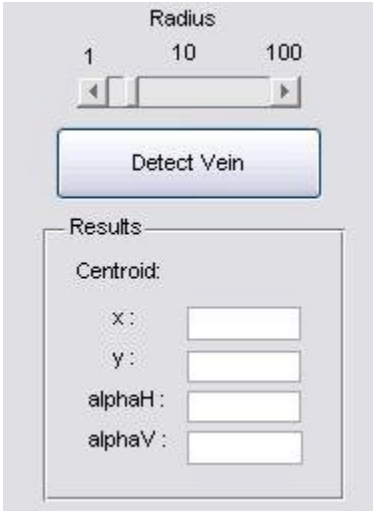


Figure 3-12. Détection de la veine

Les résultats de l'analyse apparaissent alors dans le cadre intitulé « Results ». Les valeurs 2-D du centre du cercle, et donc du centre de la veine, apparaissent dans les encadrés « x » et « y ». Pour

des fins de comparaison visuelle rapide avec les images ultrasons, les valeurs apparaissent en pixels et sont plus tard traduite en mesures métriques. De plus, les coordonnées « x » et « y » sont par rapport à la dernière image enregistrée, au point d'entrée du système aiguille-cathéter, et sont différentes des « x » et « y » tridimensionnels. En effet, elles correspondent aux « x » et « z » tridimensionnels mais à partir de la sonde ultrasons.

Les angles « alphaH » et « alphaV », qui ne sont que les inclinaisons horizontale et verticale de l'aiguille, sont aussi calculés et mis en relief une fois la veine détectée.

3.5.3 Positionnement et Insertion

Afin de valider les résultats obtenus, le bouton « Draw Vein » est mis en place pour permettre une visualisation 3-D de la veine et donc de l'insertion de l'aiguille.

L'illustration ci-dessous met en relief un exemple de « tridimensionnalité » : la veine est distinguée sous deux plans, le premier est une vue de haut (plan x-y du repère 3-D) alors que le second est une vue de côté (plan y-z du repère 3-D).

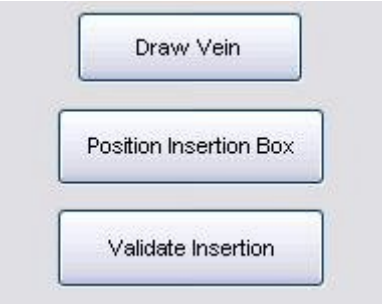


Figure 3-13. Positionnement et validation de l'insertion

Le système attend la validation de l'usager qui, basé sur les valeurs obtenues et la tridimensionnalité de la veine, prend la décision d'abandonner les résultats obtenus et répéter la simulation ou bien de procéder avec l'insertion de l'aiguille. Dans le second cas, le bouton « Position Insertion Box » envoie les informations nécessaires aux moteurs en passant par la carte Lynxmotion SSC-32, liée à l'ordinateur par un câble série RS-232. Les moteurs coordonneront leurs mouvements pour positionner le second bras robotique au point d'insertion désiré

(déterminé par la dernière image analysée par MATLAB) et ajuster le positionnement de l'aiguille et du cathéter.

Enfin, et pour cause de sécurité additionnelle, le système attend une dernière validation de la part de l'usager pour insérer l'aiguille puis le cathéter. En effet, en cliquant sur « Validate Insertion », les moteurs situés à l'extrémité du second bras robotique débutent leurs mouvements continus et insèrent l'aiguille et le cathéter légèrement dans la veine en premier lieu, puis poursuivent avec une insertion du cathéter seul d'une profondeur de 5cm, l'aiguille restant en place pour être plus tard retirée par l'utilisateur.

3.6 Second bras robotique

Avec un squelette similaire au premier, le second bras robotique est activé pour insérer l'aiguille et le cathéter dans la veine détectée. Par contre, ce bras robotique est gouverné par les lois de cinématique inverse. En se basant sur les coordonnées détectées auparavant, les positions respectives des moteurs composant le second bras robotique sont déterminées. A chaque position correspond un angle de rotation du moteur en question.

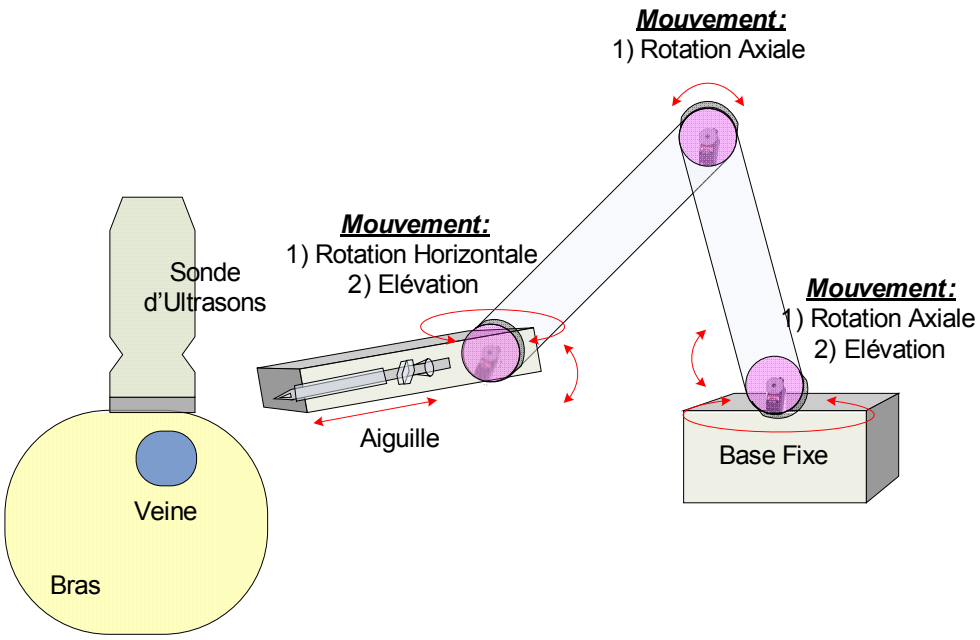


Figure 3-14. Schéma du second bras robotique: 3 moteurs Hitec et 4 moteurs 5G assurent les différents mouvements nécessaires au positionnement de l'aiguille et du cathéter, et de leur insertion subséquente dans la veine

Pour assurer une stabilité et précision, le bras est formé de 3 moteurs Hitec et 4 moteurs 5G assurant une mobilité limitée seulement par l'étendue du bras. Une coordination de ces sept moteurs, contrôlée par Matlab, aide à exécuter les tâches suivantes :

- La boite contenant l'aiguille et le cathéter est placée correctement
- L'inclinaison de la boite est ajustée
- L'aiguille et le cathéter pénètrent de 5mm la barrière de la peau
- L'aiguille est immobilisée à l'intérieur de la veine
- Le cathéter continue son chemin et glisse 10 mm dans la veine

CHAPITRE 4 ARTICLE : ULTRASOUND-GUIDED AUTOMATIC VENOUS ACCESS INSERTION *

Germain G. Aoun, B.Ing.

Institute of Biomedical Engineering, École Polytechnique de Montréal,

Montréal, QC, CANADA

E-mail: germain.aoun@polymtl.ca

Mohamad A. Sawan, PhD

Institute of Biomedical Engineering, École Polytechnique de Montréal,

Montréal, QC, CANADA

E-mail: mohamad.sawan@polymtl.ca

Thomas M. Hemmerling, MD, DEAA

Institute of Biomedical Engineering, Université de Montréal, and Department of Anesthesiology,

McGill University, Montréal General Hospital,

Montréal, QC, CANADA

Email: thomashemmerling@hotmail.com

---

* Cet article est dans l'attente d'un brevet d'invention et n'a donc pas encore été publié.

This work was performed using departmental internal funds. In part presented at the Biomedical Engineering Seminar course at Université de Montréal, Montreal, Canada, April 2008 and submitted at the Journal of Computers in November 2009.

Running Title: *Ultrasound-guided automatic venous access insertion*

Keywords: Anesthesia; robotics; ultrasound; image-guided; venous access; needle insertion

Address correspondence:

T. M. Hemmerling, MD, DEAA

Department of Anesthesiology

McGill University Health Center

Montreal General Hospital

1650 Cedar Avenue

Montréal (Québec) H3G 1A4

Canada

Ph: +1-514-934-1934 x43677

Fax: +1-514-934-8249

4.1 Abstract

—Purpose: Automation is a common concept that revolutionized multiple domains, from automobile industry to telecommunications, passing by medicine. Little automation has been introduced to the anesthesia domain. The purpose of this project is to develop an automated system for intravenous catheter insertion, and thus trigger automation use in the anesthesia domain.

—Methods: A robotic arm performs ultrasound scans on the venous site (a phantom arm), real-time snapshots are analyzed by Matlab algorithms, the vein is detected, and corresponding coordinates are sent to a second robotic arm that will position the catheter and needle bundle at the correct insertion point.

—Results: Two robotic arms were built: the first one, guided by 6 servomotors, performs automatic ultrasound scans on the venous site, and the second one, guided by 7 servomotors, positions the catheter and needle at insertion point, then inserts the needle in the vein. Analysis is performed on a Matlab graphical user interface (GUI), using Hough transform.

Index Terms—Anesthesia, Robotics, Ultrasound, Image-guided, Venous access, Needle insertion

4.2 Introduction

Anesthesiologists work in a complex environment where they monitor and record patient variables, then make patient-related decisions based on numerous monitored parameters [1]. Vigilance, parallel decision making and fine motor skills are the features that characterize the anesthesiologist in the high-risk environment of operating rooms [2]. The drug delivery process during general anesthesia is preceded by the needle and catheter insertion into the (peripheral) vein. This insertion is equally dependent on the subjective assessment of the anesthesiologist and the correct visualization of the vein.

We recently developed a novel automatic process that positions a needle-catheter bundle correctly at the site of the intended venous access, using ultrasound images of the vein area. This automation is an objective process that doesn't rely on the subjective assessment of the anesthesiologist.

The focus of this study is: 1) the development, in the anesthesia domain, of a robotic arm that performs ultrasound scans on the site, 2) the setup of a graphical user interface and algorithm that analyze the scans and detect the vein, and 3) the development of a second robotic arm that correctly positions the needle-catheter bundle and inserts the needle in the insertion point.

4.3 Methods

The algorithm representing the workflow is shown in Figure 1.

The needle/catheter insertion site is cleansed and conducting gel is applied. An ultrasound probe scans the gel area, showing the vein on a Sonosite Micromaxx portable ultrasound machine (Sonosite, Bothell, WA, USA). An s-video cable transmits the video data to a high-quality multifunctional video converter, the Canopus TwinPact100 (Canopus, San Jose, CA, USA). A firewire cable circulates the video stream to a computer where Matlab software (The Mathworks, Natick, MA, USA) projects it on the computer screen for snapshot analysis. Once the data acquisition and image treatment is finisher, snapshots of the vein are detected on each image. Image conversion, image erosion, hough transform and edge detection are combined to find corresponding vein coordinates and calculate resulting slopes, yielding the vein's position, specifically horizontal and vertical angles [3]. Using inverse kinematics, the directions to the insertion point are sent through an RS-232 serial cable to an SSC-32 chip (Lynxmotion, Pekin, IL, USA) which relays specific commands to servomotors enabling the correct positioning of the needle-catheter bundle over the desired insertion point.

4.3.1 Robotic arms

In order to ensure automation, two robotic arms were built. The system is manipulated by kinematics laws, mostly autonomous (anesthesiologist input is needed for security measures and parameter definition), targeted for anesthesia usage on any body part, and is intended to operate in or outside operating room, by an initiated personnel.

The first robotic arm ensures the locking of a L25 ultrasound probe (Sonosite, Bothell, WA, USA) and controls its successive scans on the site (in our case: the patient's arm), whereas the second one controls the needle-catheter correct positioning – three-dimensional coordinates and angles – over the insertion point, and its subsequent insertion in the vein.

Both robotic arms are sent data serially through an SSC-32 chip (Lynxmotion, Pekin, IL, USA). This transfer of information enables a set of Hitec (Hitec RCD, Poway, CA, USA) and 5G (5G,) servomotors to rotate accordingly. Servomotors are fixed with aluminum casings and connected with aluminum rods and metallic springs and screws. A weighted plastic base ensures equilibrium and smooth rotation.

4.3.2 Graphical interface

A graphical user interface was elaborated on Matlab (The Mathworks, Natick, MA, USA) software to facilitate the merging of different components and permit an easy practical approach. The interface was approved by two anesthesiologists and three engineers working in the same environment [4].

The interface is divided into three parts: An initial part initializes the first robotic arm which positions itself in place, a connection to the ultrasound shows a real-time video of the ultrasound scan on the computer screen, and snapshots are taken automatically, or based on the anesthesiologist's expertise. Previously saved ultrasound images can also be loaded. The second part deals with image analysis.

Selected snapshots are analyzed first by converting them to black and white with a color threshold of 0.75, then by applying image erosion to eliminate unwanted noise and finally using edge detection and Hough transform to detect the circle contour, i.e. the vein, and its coordinates relatively to the range of the ultrasound probe. The third and last part of the interface uses the coordinates and mechanical aluminum rods measurements as parameters for inverse kinematics. Calculations are done to find the position of each motor in space and the subsequent rotation of each motor. This data is then converted to strings of commands that are sent to the corresponding servomotors through the SSC-32. The result is a correct positioning of the second robotic arm, and thus of the needle-catheter bundle, over the insertion point. Finally, a last string of data is sent to control the insertion of the needle-catheter bundle, then the catheter only, in the correct vein location.

4.4 Results

4.4.1 Robotic arms

As shown in Fig.3, both robotic arms have similar functioning and resemble a real human arm, with a two-way shoulder rotation, elbow, wrist, and fingers movement. The first robotic arm is composed of 6 servomotors ensuring the lateral movement and elevation of the “shoulder” (two servomotors), the rotation of the “elbow” (one servomotor) and “wrist” (three servomotors) and the “finger” or “clamp” locking the ultrasound probe (one servomotor). The second robotic arm consists of 7 servomotors ensuring the lateral movement and elevation of the “shoulder” (two servomotors), the rotation of the “elbow” (one servomotor) and “wrist” (two servomotors, for horizontal and vertical alignment of the needle-catheter bundle). For both robotic arms, the base has an elevation of 13 cm, whereas aluminum rods connecting the shoulder to the elbow, and the elbow to the wrist are, respectively, 12.6 cm and 10.6 cm long. Metallic screws and springs are used to support different parts.

4.4.2 Graphical interface

The graphical user interface shown below was elaborated on Matlab. The ultrasound image snapshots are successively loaded then converted to black and white and cleansed using images erosion Matlab functions. Edge detection and Hough transform are then applied to detect the circle like shape in the image, i.e. the vein.

The vein pixel coordinates are then displayed as x and y and the horizontal (alphaH) and vertical (alphaV) angles are then consequently calculated for the current image and the one preceding it. The angles are illustrated in two longitudinal views, one from the side, emphasizing alphaV (in green, top), and one from above, emphasizing alphaH (in green, bottom).

4.4.3 Preliminary results

A phantom arm was used for model. The arm consists of a gel cylinder, on top of which a water-filled elastic tube, representing the vein and its fluids, was attached using silicon glue. Initial ultrasound scanning of the phantom arm gave similar image quality and texture for a real human arm.

Nine tests were performed at different locations. For each test five parameters were measured: alphaH, alphaV, X, Y, and Z.

alpha H and alpha V represent the horizontal and vertical angles, whereas X, Y, and Z represent the insertion point three-dimensional coordinates. For each of these parameters, the theoretical (calculated) value was saved, as well as the practical (robot positioning) value. Errors were calculated in respective units. Preliminary results are shown in Table I.

For both inclination angles, alphaH and alphaV, the error was less than one degree, whereas for X, Y, and Z coordinates, the error was noted less than 0.5 cm.

4.5 Discussion

alphaH error was of average 0.74 degrees, whereas alphaV error was of average 0.28 degrees, showing a minimal angular error less than 1 degree. X, Y, and Z average errors ranged between 0.06 and 0.32 cm.

Enhancements will be performed to stabilize the angular error, and minimize the three-dimensional coordinates' error to keep them below 0.1 cm.

4.6 Conclusion

This system is highly representative of the current increase and importance of automation in the medical field [5]. Indeed, using robotics and automotive processes as active devices to perform simple routine tasks in the operating room alleviates the medical staff's workload [6].

This automated system in itself is just one element of a larger system that helps the anesthetist carry preoperative planning, intra operative dosing and post operative procedures. By perfecting such systems, we hope to trigger advanced research and use of automotive processes as active devices in the underestimated anesthesia domain.

Table 1

Preliminary results

	1			2			3			4			5		
	T	P	≠	T	P	≠	T	P	≠	T	P	≠	T	P	≠
alphaH	11.4	11	0.4	-11.4	11	-22.4	11.2	10.7	0.5	-11.2	-10.7	-0.5	-21.7	-21.1	-0.6
alphaV	-1.27	-1.6	0.33	1.27	1.6	-0.33	1.82	1.8	0.02	-1.82	-1.8	-0.02	-0.55	-0.82	0.27
X	16.8	17.1	-0.3	17.2	17.5	-0.3	17.2	17.1	0.1	17.6	17.9	-0.3	17.5	17.8	-0.3
Y	13	13.1	-0.1	13	13.1	-0.1	13	13.2	-0.2	13	13.3	-0.3	15	15.1	-0.1
Z	-2.9	-2.8	-0.1	-2.95	-2.9	-0.05	-2.95	-2.8	-0.15	-2.9	-2.8	-0.1	-2.88	-2.9	0.02

	6			7			8			9		
	T	P	≠	T	P	≠	T	P	≠	T	P	≠
alphaH	21.7	21.1	0.6	-2.54	-3	0.46	8.4	7.6	0.8	32.5	30.1	2.4
alphaV	0.55	0.82	-0.27	2.4	3	-0.6	-1.9	-1.91	0.01	-1.14	-1.85	0.71
X	16.8	16.7	0.1	16.9	17.5	-0.6	16.4	16.1	0.3	16.4	15.9	0.5
Y	15	15.4	-0.4	15	15.2	-0.2	17	17.4	-0.4	14	14.5	-0.5
Z	-2.91	-2.9	-0.01	-2.94	-2.9	-0.04	-2.85	-2.95	0.1	-2.85	-2.9	0.05

Mean	StDv
0.7400	0.6347
0.2844	0.2501
0.3111	0.1616
0.2556	0.1509
0.0689	0.0459

4.7 Legends

Figure 1: General algorithm

The figure shows the general algorithm of the project

- Site is prepared (usually conductive gel is applied)
- Ultrasound transducer scans the site area
- Data is collected
- Data is analyzed
- Vein is detected
- Needle and catheter bundle are inserted in the site area

Figure 2: Graphical User Interface (GUI) developed on Matlab

The figure shows the graphical user interface (GUI) that was developed on Matlab.

Figure 3: On the right: First robotic arm with 6 motors for movement (6 degrees of freedom). On the left: Second robotic arm with 5 motors for movement (5 degrees of freedom) and 2 motors for needle/catheter insertion

Figure 1

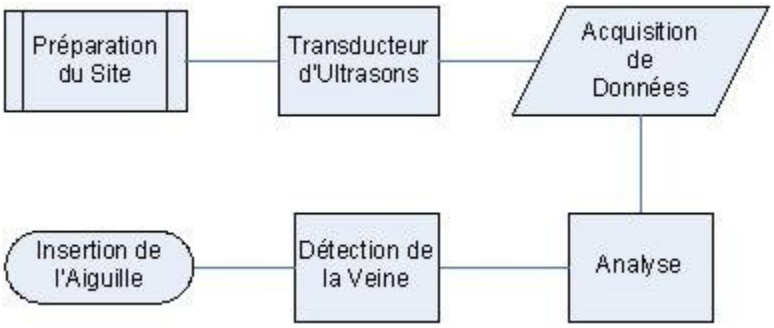

Préparation du Site
Transducteur d'Ultrasons
Acquisition de Données
Insertion de l'Aiguille
Détection de la Veine
Analyse

Figure 2

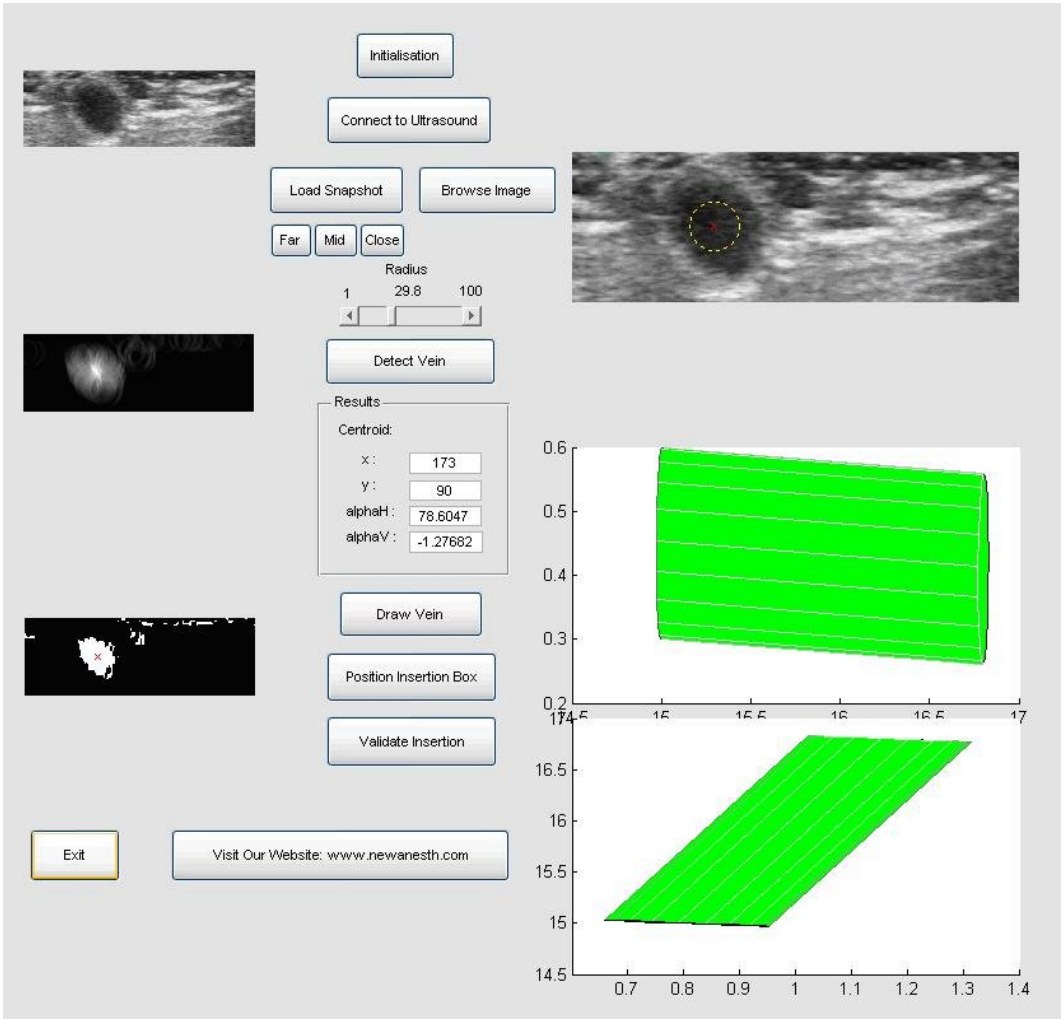
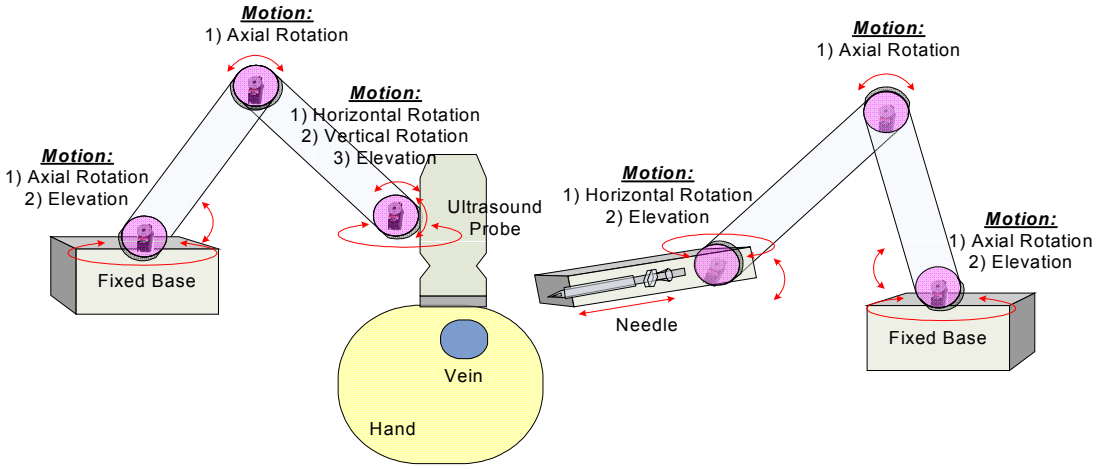

Initialisation
Connect to Ultrasound
Load Snapshot
Browse Image
Far
Mid
Close
Radius
1
29.8
100
Detect Vein
Results
Centroid:
x :
173
y :
90
alphaH :
78.6047
alphaV :
-1.27682
Draw Vein
Position Insertion Box
Validate Insertion
Exit
Visit Our Website: www.newanesth.com
0.6
0.5
0.4
0.3
0.2
16.5
16
15.5
15
14.5
0.7
0.8
0.9
1
1.1
1.2
1.3
1.4

Figure 3


Motion:
1) Axial Rotation
Motion:
1) Axial Rotation
2) Elevation
Motion:
1) Horizontal Rotation
2) Vertical Rotation
3) Elevation
Ultrasound Probe
Fixed Base
Vein
Hand
Motion:
1) Axial Rotation
Motion:
1) Horizontal Rotation
2) Elevation
Needle
Motion:
1) Axial Rotation
2) Elevation
Fixed Base

4.8 References

[1] McDonald JS, Dzwonczyk R, Gupta B, Dahl M. A second time-study of the anaesthetist's intraoperative period. Br.J.Anaesth. 1990 May;64(5):582-5.

[2] McDonald JS, Dzwonczyk RR. A time and motion study of the anaesthetist's intraoperative time. Br.J.Anaesth. 1988 Dec;61(6):738-42.

[3] Duda, R. O. and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," Comm. ACM, Vol. 15, pp. 11–15 (January, 1972)

[4] Blike GT, Surgenor SD, Whalen K. A graphical object display improves anesthesiologists' performance on a simulated diagnostic task. J.Clin.Monit.Comput. 1999 Jan;15(1):37-44.

[5] Dario, P., Guglielmelli, E., Allotta, B., Robotics in medicine, Intelligent Robots and Systems '94. 'Advanced Robotic Systems and the Real World', IROS '94. Proceedings of the IEEE/RSJ/GI International Conference on, 1994, Vol.2, p.739-752

[6] Harold H. Kraft, David Eric Lees, Closing the loop: how near is automated anesthesia? Southern medical journal, vol.77, No.1, January 1984

CHAPITRE 5 DISCUSSION GÉNÉRALE

Le développement de ce projet a été exécuté en deux parties enchevauchées. Tout d'abord, un code Matlab a été élaboré en se basant sur la transformée de Hough pour déterminer la présence de cercles dans une image ultrasonographique quelconque. L'image est premièrement convertie en noir et blanc pour une meilleure analyse selon un indice de conversion, qui doit être bien choisi pour transmettre suffisamment d'information à l'analyse. Puis la transformée est appliquée et les cercles sont détectés. Une interface graphique a été développée pour offrir une meilleure interaction avec l'utilisateur et lui permettre de mieux visualiser les avancements. La seconde partie du projet consiste en la construction mécanique de deux bras robotiques, le premier aidant le mouvement de la sonde ultrasonographique au-dessus du bras, et le second positionnant et insérant l'aiguille et le cathéter au site d'injection.

Au cours de l'avancement du projet, ces deux parties se sont chevauchées à différentes reprises. En effet, l'algorithme doit être codé au fur et à mesure que les bras mécaniques sont bâtis. Les dimensions et emplacements des bras, ainsi que leurs liaisons et joints, doivent être insérés dans l'algorithme. De plus, les limitations de l'algorithme régissent le positionnement des bras, ainsi que le matériel utilisé (moteurs hitec, 5G, SSC-32, etc). Etant donné qu'il n'y a aucune base ou revue de littérature portant sur un sujet pareil, la meilleure façon de procéder était par tâtonnement analytique : une simple analyse mécanique ou informatique est faite, puis les différents choix sont utilisés au hasard afin d'évaluer les résultats (type et puissance des moteurs, longueur des bras, matériel utilisé, méthode de codage, etc.)

Ce projet s'est vu decerner le prix « EXCELLENCE IN TECHNOLOGY INNOVATION AWARD », offert par la SOCIETY OF TECHNOLOGY IN ANESTHESIA, au cours de leur rencontre annuelle, le 16 janvier 2009.

5.1 Résultats

Afin de bien analyser les résultats, un plan de tests a été élaboré comme suit :

1. Le premier bras robotique place la sonde ultrasons au-dessus du bras fantôme, à une distance prédéfinie ($X_1 = 15$ *cm,* $X_2 = 13.5$ *cm puis* $X_3 = 16.5$ *cm*) de la base du bras

robotique. L'utilisateur doit s'assurer de la présence d'une quantité suffisante de gel conducteur et de la clarté de l'image avant de procéder.

2. Un balayage automatique de *0.5 cm* a lieu, avec une prise de 2 images ou plus. L'analyse Matlab a lieu sur les images enregistrées afin de détecter la veine dans chaque image, et déduire la possibilité d'injection, c'est-à-dire si la veine est droite (si les deux détections sont proches) ou crochue (si les deux détections sont éloignées). Si la veine est droite, les résultats sont validés et les coordonnées relatives enregistrées. Dans le cas contraire, la veine est rejetée et l'utilisateur devra choisir un nouvel emplacement de la sonde ultrasonographique.
3. Les coordonnées sont transmises en mouvement de moteurs contrôlés par les lois de cinématique. Les moteurs se positionnent pour placer l'aiguille et le cathéter au-dessus, puis à l'intérieur de la veine ciblée, après l'avoir transpercée.

Pour en valider le fonctionnement, ces trois étapes successives ont été testées à maintes reprises. Ainsi :

1. Le positionnement du premier bras robotique au site désiré a été effectué 10 fois pour chaque position ($X_1 = 15$ *cm,* $X_2 = 13.5$ *cm puis* $X_3 = 16.5$ *cm*). À une vitesse stable et modérée, la position atteinte était toujours la même pour chaque groupe d'essais. Les moteurs étant relativement petits et faibles, leurs têtes étant en plastique, et leurs liens en minces tiges métalliques, un déplacement rapide peut forcer le mouvement et modifier le positionnement. Donc, pour des mouvements à vitesse modérée, la position du premier bras robotique est la même pour les essais appartenant à un groupe ($X_1 = 15$ *cm,* $X_2 = 13.5$ *cm puis* $X_3 = 16.5$ *cm*).

Tableau 5.1. Résultats des tests de positionnement du premier bras robotique

X_1 = 15 cm	*Essai*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
	Position	15.3	15.1	14.8	14.6	15.1	14.9	15.1	15.3	14.8	15.3

X_2 = 13.5 cm	*Essai*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
	Position	13.5	13.6	13.2	13.8	13.5	13.4	13.7	13.4	13.3	13.4

X_3 = 16.5 cm	*Essai*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
	Position	16.2	16.4	16.4	16.2	16.3	16.4	16.3	16.4	16.2	16.4

Il est important de noter que non seulement le poids de la sonde ultrasonographique est considéré important par rapport au support mécanique le guidant, mais aussi les supports formant le bras mécanique sont en aluminium et les joints en plastique. Ceci limite la précision de position et permet une marge d'erreur due au jeu de l'élasticité et manque de rigidité de l'aluminium et du plastique utilisés.

2. En ce qui concerne la détection de la veine dans les images, dix autres tests ont été effectués sur dix images ultrasonographiques différentes (differents emplacements sur le bras fantôme, et meme sur un bras humain). Les résultats étaient 100% corrects pour six de ces images, alors qu'ils étaient complètement erronés pour quatre autres images. Pour les détections correctes, l'image était de bonne qualité et ne montrait qu'une seule veine.

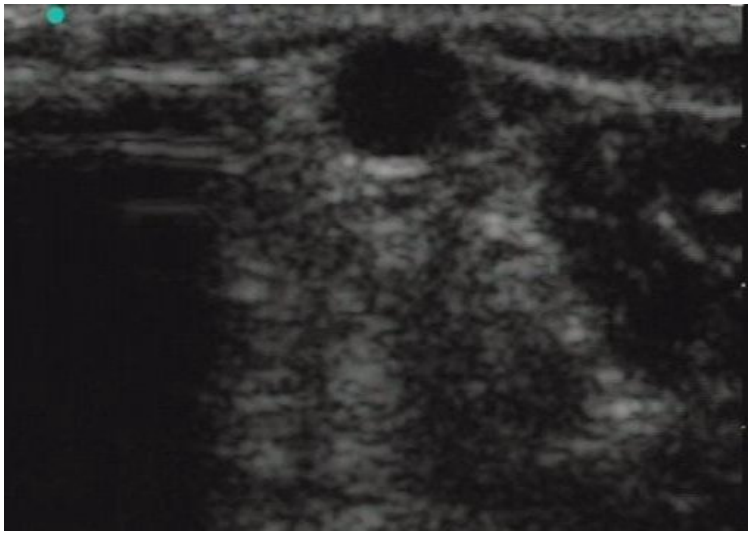


Figure 5-1. Image de bonne qualité, montrant une seule veine

Alors que pour les détections erronées, l'image était soit de mauvaise qualité soit comprenait plusieurs grosses veines. L'algorithme présentement élaboré est incapable de faire la différence entre les différentes veines (en forme de gros cercles sur l'image) et de choisir celle désirée par l'utilisateur.

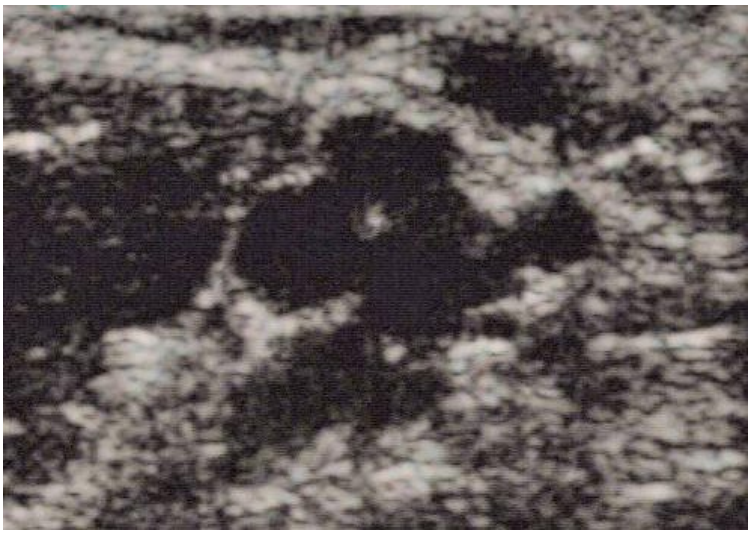


Figure 5-2. Image de mauvaise qualité et montrant plusieurs veines

3. Finalement 10 tests globaux ont été effectués pour le positionnement final de l'aiguille et du cathéter. Les résultats de 9 de ces 10 tests ont été enregistrés, le dernier ayant été jugé comme artefact dû à la faible mémoire vive et la lenteur des calculs faits par l'ordinateur. Cinq paramètres étaient notés : les coordonnées X, Y, et Z de la pointe de l'aiguille (valeur théorique visée, et valeur pratique atteinte), et les angles d'inclinaisons horizontal et vertical. Ces données ont été classées en une colonne théorique (résultats de l'analyse) notée « T » et une autre pratique (mouvement du second bras robotique et positionnement réel de l'aiguille) notée « P ». La différence entre ces deux colonnes en a été déduite pour enfin calculer la moyenne et la déviation standard pour chacun des ces cinq paramètres. Ainsi, dans le tableau suivant, T représente la valeur théorique visée, P la valeur pratique atteinte, ≠ la différence entre ces deux valeurs, alphaH et alphaV les angles d'inclinaisons horizontale et verticale de la pointe de l'aiguille (en degrés), X, Y, et Z, les coordonnées de la pointe de l'aiguille (en cm), Mean la moyenne et StDv l'écart-type.

Tableau 5.2. Résultats de 9 tests.

	1			2			3			4			5		
	T	P	≠	T	P	≠	T	P	≠	T	P	≠	T	P	≠
alphaH	11.4	11	0.4	-11.4	11	-22.4	11.2	10.7	0.5	-11.2	-10.7	-0.5	-21.7	-21.1	-0.6
alphaV	-1.27	-1.6	0.33	1.27	1.6	-0.33	1.82	1.8	0.02	-1.82	-1.8	-0.02	-0.55	-0.82	0.27
X	16.8	17.1	-0.3	17.2	17.5	-0.3	17.2	17.1	0.1	17.6	17.9	-0.3	17.5	17.8	-0.3
Y	13	13.1	-0.1	13	13.1	-0.1	13	13.2	-0.2	13	13.3	-0.3	15	15.1	-0.1
Z	-2.9	-2.8	-0.1	-2.95	-2.9	-0.05	-2.95	-2.8	-0.15	-2.9	-2.8	-0.1	-2.88	-2.9	0.02

	6			7			8			9			Mean	StDv
	T	P	≠	T	P	≠	T	P	≠	T	P	≠		
alphaH	21.7	21.1	0.6	-2.54	-3	0.46	8.4	7.6	0.8	32.5	30.1	2.4	0.7400	0.6347
alphaV	0.55	0.82	-0.27	2.4	3	-0.6	-1.9	-1.91	0.01	-1.14	-1.85	0.71	0.2844	0.2501
X	16.8	16.7	0.1	16.9	17.5	-0.6	16.4	16.1	0.3	16.4	15.9	0.5	0.3111	0.1616
Y	15	15.4	-0.4	15	15.2	-0.2	17	17.4	-0.4	14	14.5	-0.5	0.2556	0.1509
Z	-2.91	-2.9	-0.01	-2.94	-2.9	-0.04	-2.85	-2.95	0.1	-2.85	-2.9	0.05	0.0689	0.0459

5.2 Comparaison avec d'autres études

Le développement des deux parties composant ce projet (partie algorithme et partie mécanique) est unique en son genre et une première mondiale. Cependant, bien qu'aucune autre étude ne traite spécifiquement de l'insertion automatique guidée par ultrasons, un rapprochement sera fait

avec quelques études similaires utilisant des images ultrasonographiques pour effectuer une insertion, ou placement automatique d'une aiguille pour une tâche quelconque.

Ainsi, bien que plusieurs algorithmes d'extraction de critères d'images ultrasonographiques aient été proposés dans la littérature [62][63][64][65], à moins d'utiliser des équipements informatiques de pointe, la vitesse d'analyse d'images ultrasonographiques est de l'ordre des secondes, allant jusqu'aux dizaines de secondes [66][67][68]. Dans le cas du présent projet, l'analyse d'images se faisait en moins d'une dizaine de secondes, dépendamment de la qualité de l'image enregistrée.

La majorité des études comparativement proches de la présente utilisent les mêmes informations dans leurs interfaces graphiques (GUI – Graphical User Interface). La veine détectée, ainsi que la visualisation en trois dimensions sont omniprésentes, alors que d'autres informations varient d'une étude à l'autre; certaines illustrent l'image initiale, d'autres les images intermédiaires, d'autres encore les coordonnées, etc. L'interface graphique élaborée au cours de ce mémoire offre à l'utilisateur :

- le choix entre une sélection manuelle ou automatique des images, ou vidéo, à analyser,
- le contrôle des moteurs gérant le mouvement de la sonde ultrasonongraphique,
- la visualisation des images initiales, ou différentes coupes prises de la vidéo,
- l'image après traitement de couleur (noir et blanc),
- l'application de la transformée de Hough,
- l'image avec la veine détectée,
- les coordonnées du point d'insertion de l'aiguille,
- la représentation en trois dimensions (sous forme de deux images bidimensionnelles) de la veine détectée,
- la validation des résultats pour l'enclenchement subséquent de l'insertion de l'aiguille

L’interface graphique élaborée offre donc à l’utilisateur plus de fonctionnalités permettant ainsi une insertion automatisée et vérifiée par l’expert médical.

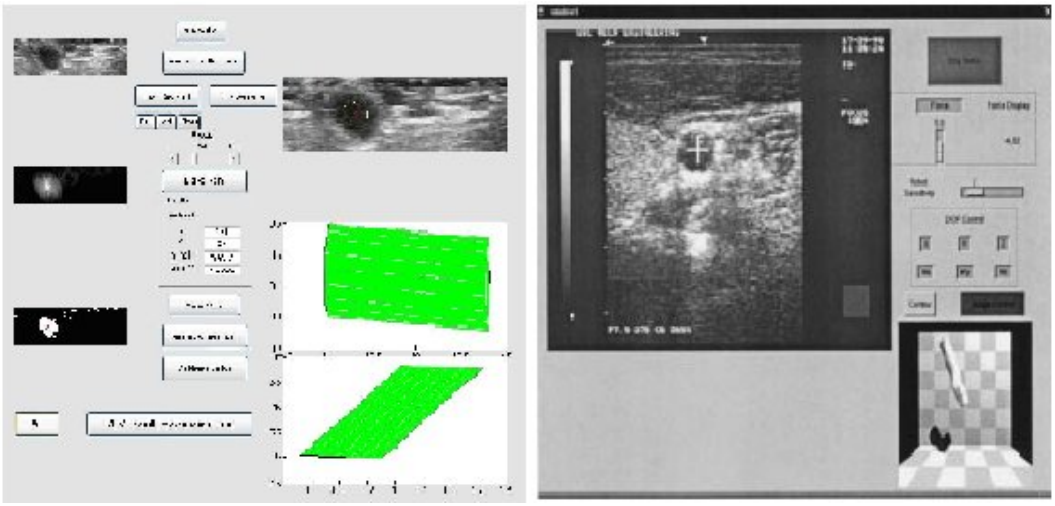


Figure 5-3. Comparaison entre l'interface graphique developpée au cours de ce projet , et une autre interface developpée pour une étude similaire

En référence à la revue de littérature développée dans la section 2.5.1 de ce document, l’interface graphique résultante de ce projet respecte les caractéristiques types d’une interface graphique, notamment les informations présentes sont consistantes, leur disposition visuelle est esthétique et permet une analyse directe, les métaphores ou vocabulaire utilisés sont clairs et concis, et enfin, l’interface est centrée sur l’utilisateur et répond à ses besoins en montrant l’information requise.

La visualisation 3-D de la veine représente une partie importante pour la validation initiale de la détection de veine. En effet, cette étape préliminaire permet à l’utilisateur de s’assurer rapidement si la veine est linéaire dans les alentours de la région d’insertion, et donc adéquate à une pénétration d’aiguille, ou croche, et donc inadéquate.

Puisque la localisation de la sonde ultrasonographique est connue, cette information peut être jumelée à celle déterminée à partir des images analysées afin de reconstruire une visualisation en trois dimensions à partir d’un minimum de deux images en deux dimensions. L’algorithme développé au cours de ce projet a été comparé à deux autres algorithmes similaires en termes de

besoin d'informations de base et de rapidité de reconstruction : l'outil d'acquisition et de visualisation d'images ultrasonographiques 3-D « Stradx » [69] et l'algorithme d'extraction de contours « Star-Kalman » [70]. Bien que « Stradx » et « Star-Kalman » soient plus précis et offrent une reconstruction graphique en trois dimensions, les résultats de l'algorithme décrit au cours de la présente sont comparables et offrent une analyse doublement bidimensionnelle permettant une meilleure connaissance des dimensions de la veine et de son orientation par rapport à la sonde ultrasonographique.

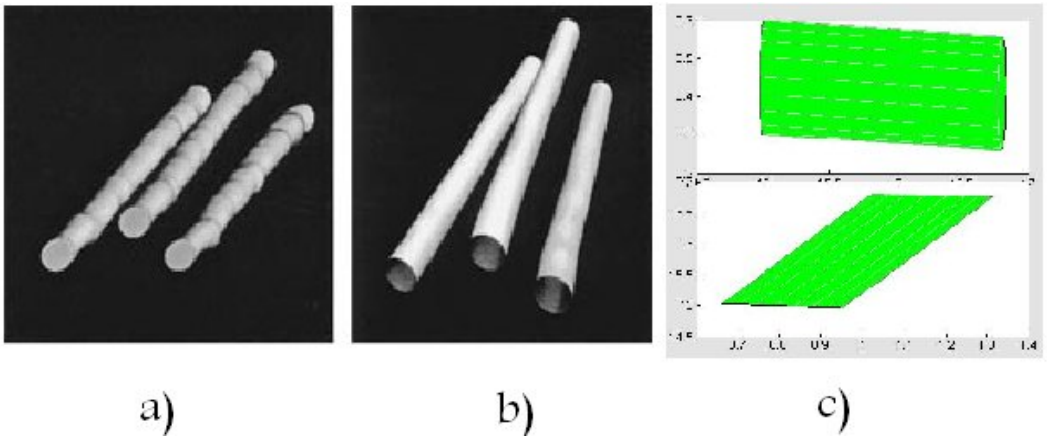


Figure 5-4. Reconstruction partielle 3-D de l'image détectée par sonde ultrasonographique en utilisant: a) l'outil d'acquisition d'images 3-D Stradx, b) l'algorithme Star-Kalman pour extraction de contours, c) l'algorithme développé au cours de ce projet

En ce qui concerne l'orientation mécanique de l'aiguille, certaines études restreignent les mouvements à un axe vertical uniquement [71], d'autres immobilisent l'aiguille et se concentrent plus sur la détection de la veine [72], d'autres développent un système avec 3 degrés de liberté pour le mouvement de la sonde avec uniquement 1 degré de liberté pour le positionnement de l'aiguille [73], et d'autres enfin construisent leur prototype avec 6 degrés de liberté pour assurer un mouvement complet [74]. Quels que soient les cas, la calibration du positionnement de l'aiguille consume un temps pouvant aller jusqu'à une heure.

Côté mécanique, le projet présenté au cours de ce document offre :

- 7 degrés de liberté pour le premier bras robotique : à travers 7 moteurs réglant le mouvement; 3 pour le positionnement du bras, 3 pour le placement de la sonde ultrasonographique, et 1 pour capter la sonde en place,
- 7 degrés de liberté pour le second bras robotique : à travers 7 moteurs réglant le mouvement; 3 pour le positionnement du bras, 2 pour le placement de l'aiguille et du cathéter, et 2 pour insérer l'aiguille puis le cathéter dans la veine,
- Une calibration et mouvement mécaniques dans l'ordre des secondes, indépendamment des dimensions des bras, de l'aiguille, ou du cathéter, le tout étant facilement réglable à partir du code Matlab.

Okazawa et al. [75] se basent sur des images ultrasons pour manier un appareil à pointe amovible permettant un positionnement adéquat d'une aiguille pour des interventions cliniques cutanées. Leur projet et celui présenté tout au long de ce mémoire se rejoignent dans deux étapes importantes : Premièrement, l'emplacement de la veine et le point d'insertion de l'aiguille sont déterminés à travers l'image ultrasonographique du site en question. Deuxièmement, la direction d'orientation de l'aiguille et son positionnement sont déterminés de même à travers une analyse de l'image ultrasonographique.

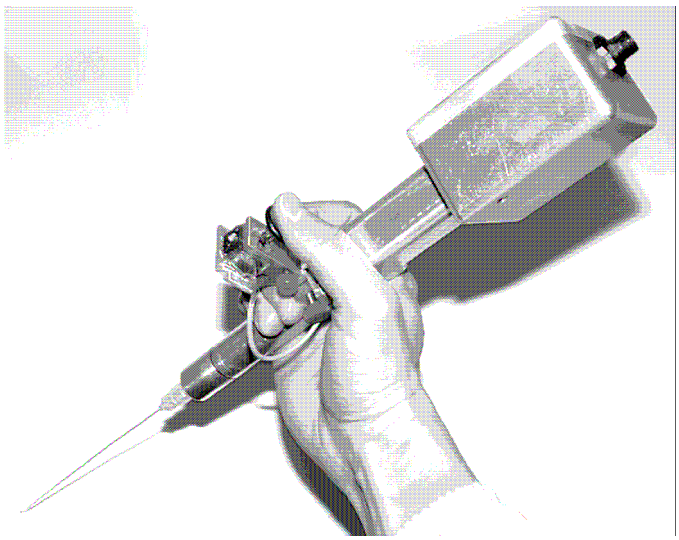


Figure 5-5. Guidance manuelle et par contrôle télécommandé

Cependant, Okazawa et al. visualisent l'image ultrasonographique et déterminent la veine visuellement pour ensuite placer le bras robotique manuellement et positionner l'aiguille à l'aide d'une télécommande. Le positionnement n'est donc pas automatique. Par contre, dans notre projet, une fois la veine détectée, le positionnement du second bras robotique a lieu de façon automatique, sans intervention humaine. Les coordonnées sont traduites en langage compris par les moteurs, puis sont envoyées au système de moteurs pour que ces derniers agencent leurs mouvements respectifs et placent l'aiguille et le cathéter au site d'insertion.

De plus, pour Okazawa et al, une erreur de positionnement de l'aiguille ou une mauvaise interprétation peut se corriger en cours de mouvement puisque le système est contrôlé à distance par télécommande, alors que ceci n'est pas le cas pour le projet présenté dans ce rapport. En effet, en cas de mauvais calcul, aucune correction ne peut avoir lieu et la démarche entière doit être reprise du début. Alors qu'en cas de mauvais positionnement, la commande peut être envoyée de nouveau pour une correction de mouvement. Le mouvement sera ainsi répété pour placer le bras robotique à destination désirable avec une erreur généralement moins de 1mm en coordonnées, et moins de 1 degré d'erreur d'angles de positionnement.

Comparativement, Ayadi et al [74] performent des biopsies animales en utilisant un système de détection de l'emplacement de l'aiguille à partir d'images ultrasons, en combinant la transformée de Hough et la détection d'arrêtes Canny. Cette combinaison de techniques leur permet de trouver les lignes délimitant le bout de l'aiguille et de déterminer les coordonnées correspondantes, avec une marge d'erreur de moins de 1mm. Ayadi et al affirment qu'une biopsie robotique animale a déjà été proposée par certaines études, mais qu'aucune insertion robotique d'aiguille ne fut développée pour besoins d'analyses.

5.3 Limitations et améliorations

Bien que les résultats soient assez positifs, le présent projet est encore en phase prototype. Plusieurs limitations et améliorations peuvent être proposées et mises en place, mais le développement ultime du projet serait son jumelage avec McSleepy [29]. Ceci offrira un système d'anesthésie robotique futuriste mettant en jeu l'insertion automatique de l'accès veineux guidée

par ultrasons suivie par un système en boucle fermée pour l'injection automatique de 3 agents anesthésiants. Afin d'effectuer ce jumelage, certains changements doivent être entrepris dans les deux projets afin de concilier les algorithmes respectifs et l'échange de données sur une même plateforme et à travers une même interface graphique. Dans l'attente de ce développement prochain, la présente section dévoile certaines limitations du prototype actuel et les répartit selon leur aspect informatique ou mécanique, tout en offrant des améliorations possibles quand c'est le cas.

5.3.1 Aspect informatique :

L'aspect informatique du projet comporte deux limitations majeures empêchant un système plus autonome :

- Premièrement, l'initialisation du premier bras est prédéfinie dans le code ; c'est-à-dire les coordonnées initiales de l'emplacement de la sonde d'ultrasons doivent être prédéfinies manuellement dans le code Matlab. Tout changement dans l'emplacement du bras du patient ou du bras mécanique doit être suivi d'une modification correspondante dans le code Matlab. Ceci peut être corrigé par l'ajout de senseurs de position qui, une fois programmés, pourront détecter les variations de position et corriger automatiquement les données relatives dans l'algorithme Matlab.
- Deuxièmement, la transformée de Hough est écrite de façon à détecter uniquement un cercle, et donc une veine, qui apparait dans la vidéo et ne peut offrir la possibilité de visualiser plusieurs cercles (veines) afin d'en choisir la meilleure. C'est pour cette raison que les tests ont été effectués sur un bras fantôme contenant uniquement une seule veine. L'algorithme devrait être modifié afin de détecter plusieurs veines et attendre la confirmation de l'utilisateur quant au choix de la veine d'insertion.

Aussi, le prototype comporte diverses autres limitations mineures et requiert ainsi plusieurs améliorations informatiques pour offrir son potentiel maximal :

- Le port série permettant les communications entre les moteurs et le logiciel est prédéfini et toute modification doit s'ensuivre dans le code. Une amélioration possible serait la détermination automatique de ce port suite à un balayage de tous les ports disponibles et libres. Matlab pourra ainsi automatiquement déterminer quel port utiliser pour communiquer avec les moteurs.
- Certaines valeurs cibles ont été préférées après divers essais et pourraient être mieux choisies. Elles pourraient offrir une meilleure qualité d'analyse. Ex. : Y = im2bw(256-I,0.75); La valeur 0.75 a été choisie après plusieurs tests et essais, mais cela n'empêche pas de trouver une meilleure valeur.
- Les images fournies par l'appareil d'ultrasons sont rognées automatiquement afin d'éliminer le contour et de faciliter l'analyse. Ce rognage peut effacer certaines données importantes comme la date, la profondeur, le degré de précision, le mode utilisé, etc. et peut cacher des informations relatives à la veine à détecter quand celle-ci n'est pas centrée sur l'image, c'est-à-dire quand la sonde ultrasonographique n'est pas bien positionnée au centre du bras fantôme.
- La stabilité informatique peut être améliorée en ajoutant à l'interface graphique de l'utilisateur des alertes empêchant la tenue de certaines actions simultanées ou obligeant un ordre précis des actions. Un mauvais ordre de commandes peut résulter en des erreurs Matlab.
- L'algorithme, comme tout autre algorithme de prototype, peut être optimisé en modifiant les lignes de code et remplaçant certaines commandes, ou groupes de commandes, en d'autres plus performantes.

5.3.2 Aspect mécanique :

Similairement à l'aspect informatique, l'aspect mécanique du projet comporte deux limitations majeures empêchant un système plus performant. Cependant, l'aspect mécanique tel que développé n'empêche pas l'autonomie du système, mais affecte plutôt sa stabilité et précision :

- Premièrement, pour respecter le budget alloué au prototype, les tiges métalliques et les moteurs utilisés n'offrent pas une précision parfaite. Le positionnement est donc affecté par le degré d'extension du bras robotique et le poids qui y est présent. Ainsi, un bras étendu sera moins précis qu'un bras plié, et un bras simple sera plus précis qu'un bras portant du poids (probe ultrasons, système aiguille-cathéter). Une amélioration de précision peut être apportée en choisissant une meilleure qualité de moteurs et tiges métalliques pour construire le système mécanique.
- Deuxièmement, la complexité des divers degrés de liberté des deux bras robotique ajoute à l'imprécision mécanique du prototype et permet plusieurs erreurs de calculs (approximations dans l'algorithme). Vu que l'importance du système réside dans sa tridimensionnalité, cette limitation ne peut être modifiée, mais les imprécisions peuvent seulement être limitées par une meilleure qualité de composantes de système mécanique (robots et tiges métallique notamment).
- Finalement, le système peut être dédoublé afin d'offrir la possibilité de détecter deux veines et d'insérer un groupe aiguille-cathéter dans chacune. Ceci permet l'utilisation d'une veine pour l'anesthésie, et d'une autre pour les fluides. Bien que ceci nécessite des accès veineux multiples et sécuritaires, un système optimal peut être développé pour offrir cette amélioration.

CONCLUSION

L'ère présente multiplie ses efforts pour intégrer les applications robotiques en médecine et plusieurs avantages sont mis en relief; mais cependant, cet avancement est retardé par des doutes de performance, de sécurité et de coût. Bien que diverses compagnies se soient lancées dans ce domaine et aient développé de nouvelles applications robotiques médicales, le nombre total de « robots » installés dans le domaine médical demeure très bas et le marché se développe lentement. L'importance de la présence robotique dans ce domaine n'a pas encore été démontrée et les bénéfices n'ont pas encore été mis en relief.

Au cours de ce mémoire de maitrise, un aspect de l'anesthésie robotique, ou automatique, encore inexploré et offrant toute une nouvelle vision quant à l'importance des applications robotiques médicales, a été mis en relief : « L'insertion automatique d'un cathéter veineux guidée par ultrasons ».

Un bras robotique guide une sonde ultrasonographique qui balaye le bras du patient puis envoie les images à un algorithme Matlab. Ce dernier analyse différentes coupes, y détecte la veine et valide l'insertion. Les coordonnées du point d'insertion sont ensuite envoyées à un second bras robotique pour positionner l'aiguille et le cathéter, puis les insérer dans la veine étudiée. Les résultats sont consistants et encourageants, l'erreur linéaire étant de moins de 5mm et l'erreur angulaire ne dépassant pas 1 degré. Ce projet n'a pas pour but de remplacer l'anesthésiste, mais plutôt d'améliorer les capacités de l'anesthésiste en lui offrant la possibilité de se concentrer sur les soins directement relatifs au patient. Une nouvelle page a été ouverte dans l'application robotique médicale, et l'anesthésie offre des possibilités jusque-là encore inexplorées. En se basant sur les résultats du prototype de ce mémoire, on ne peut qu'affirmer que l'équipe formée par l'anesthésiste et le système automatisé fera des merveilles et offrira au patient des soins de première classe.

BIBLIOGRAPHIE

[1] Prys-Roberts, C., *Anesthesia : A practical or impractical conduct?*, Br J Anaesth, 1987; 59; 1341-5

[2] Gordon, B.J., *Medicine throughout Antiquity*. Arch Ophthal. 1949;42(2):224

[3] http://www.anesthesia-nursing.com/ormorton.jpg, Accédé en Juillet 2006

[4] *Continuum Of Depth Of Sedation Definition Of General Anesthesia And Levels Of Sedation/Analgesia*», American Society of Anesthesiologists, ASA, 2004-10-27

[5] Collier, D., *The service sector revolution : the automation of services*, Long range planning, vol.16 No.6, 1983

[6] Reimer, J., *A History of the GUI,* http://arstechnica.com/, Accédé en Janvier 2008

[7] http://toastytech.com/guis/guitimeline.html, Accédé en Janvier 2008

[8] Dario, P., Guglielmelli, E., Allotta, B., *Robotics in medicine*, Proceedings of the IEEE/RSJ/GI International Conference on, 1994, Vol.2, p.739-752

[9] Kwoh, Y.S., Hou, J., Jonckheere, E.A., Hayati, S., *A robot with improved absolute positioning accuracy for CT guided stereotactic brain surgery*, IEEE transactions on biomedical engineering, 1988, Vol.35, pp. 153-160

[10] Cleary, K., Nguyen, C., *State of the arts in surgical robotics : Clinical Applications and Technology Challenges*, Computer Aided Surgery, 2001

[11] Chapman, T., *Automation on the move*, NATURE, 2003, Vol. 421

[12] Brennan, T.A., Leape, L.L., Laird, N.M., *Incidence of adverse events and negligence in hospitalized patients*, N Engl J Med 1991;324:370–6.

[13] Kohn, L.T., Corrigan, J.M., Donaldson, M.S., *To err is human: building a safer health system*, Washington, DC: National Academy Press, 1999

[14] Taylor, R., *Medical Robotics in Computer-Integrated Surgery*, IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION, Vol. 19, No. 5, 2003

[15] Woo, R., Le, D., Krummel, T.M., Albanese, C., *Robot-assisted pediatric surgery*, The American Journal of Surgery 188 (Suppl to October 2004) 27S–37S

[16] Dario, P., Guglielmelli, E., Allotta, B., Carrozza, M.C., *Robotics for Medical Applications*, IEEE Robotics &Automation Magazine, 1996

[17] Cirac, T., *A project report on medical robotics*, Istanbul Technical University, 2006

[18] Satava, R. *Cybersurgery: Advanced Technologies for Surgical Practice*, John Wiley & Sons, Inc: NewYork, 1997.

[19] Vitez, T.S., Wada, R., Macario, A., *Fuzzy logic: theory and medical applications*, J.Cardiothorac.Vasc.Anesth. 1996 Oct;10(6):800-8

[20] Kraft, H.H., Lees, D.E., *Closing the loop: how near is automated anesthesia?* , Southern medical journal, 1984, Vol.77, No.1

[21] Merry, A.F., Webster, C.S., Mathew, D.J., *A New, Safety-Oriented, Integrated Drug Administration and Automated Anesthesia Record System*, Anesth Analg 2001;93:385–90

[22] Struys, M., De Smet, T., Mortier, E., *Closed-loop control of anaesthesia, Current opinion in anaesthesiology*, 2002, Vol. 15(4), pp 421-425

[23] Simanski, O., Kaehler, R., Schubert, A., Janda, M., Bajorat, J., Hofmockel, R., Lampe, B.P., *Automatic drug delivery in anesthesia – the design of an anesthesia assistant system*, Proceedings of the 17th World Congress, The International Federation of Automatic Control, Seoul, Korea, July 6-11, 2008

[24] O'Hara, D.A., Bogen, D.K., Noordergraaf, A., *The use of computers for controlling the delivery of anesthesia*, Anesthesiology 1992; 77 : 563-81

[25] Absalom, A.R., Sutcliffe, N., Kenny, G.N., *Closed-loop control of anesthesia using bispectral index,* Anesthesiology, 96(1):67–73, 2002

[26] Mortier, E., Struys, M., De Smet, T., Versichelen, L., Rolly, G., *Closed-loop controlled administration of propofol using bispectral analysis*, Anesthesia 1998; 53 : 749-54

[27] Westenskow, D.R., Zbinden, A.M., Thomson, D.A., Kohler, B., *Control of end-tidal halothane concentration. Part A : Anaesthesia breathing system and feedback control of gas delivery*, Br J Anaesth 1986; 58 : 555-62

[28] Charabati, S., Dubois, B., Bracco, D., Mathieu, P.A., Hemmerling, T.M., *Performance of a novel closed-loop propofol system*, proceedings of the Society for tech in Anesthesia (STA 2008) annual meeting, january 16-19, 2008, San Diego, CA, USA

[29] Canadians create anesthesia system dubbed McSleepy, http://www.cbc.ca/health/story/2008/05/02/anesthesia-system.html, Accédé en Mai 2008

[30] Okazawa, S., Ebrahimi, R., Chuang, J., Salcudean, S.E., Rohling, R., *Hand-Held Steerable Needle Device*, IEEE/ASME Transactions on Mechatronics, 2005, Vol.10, NO. 3.

[31] Cleary, K., Watson, V., Lindisch, D., Taylor, R.H., Fichtinger, G., Xu, S., White, C.S., Donlon, J., Taylor, M., Patriciu, A., Mazilu, D., Stoianovici, D., *Precision placement of instruments for minimally invasive procedures using a ''Needle Driver'' robot*, Int J Medical Robotics and Computer Assisted Surgery 2005;1(2):40–47

[32] Hong, J., Dohi1, T., Hashizume, M., Konishi, K., Hata, N., *An ultrasound-driven needle-insertion robot for percutaneous cholecystostomy,* Phys. Med. Biol. 49 (2004) 441–455

[33] Marhofer, P., Greher, M., Kapral, S., *Ultrasound guidance in regional anesthesia,* British Journal of Anaesthesia 94 (1): 7–17 (2005)

[34] Greher, M., Retzl, G., Niel, P., Kamholz, L., Marhofer, P., Kapral, S., *Ultrasonographic assessment of topographic anatomy in volunteers suggest a modification of the infraclavicular vertical brachial block*, Br J Anaesth 2002; 88: 632–6

[35] Peterson, M.K., *Ultrasound-guided nerve blocks*, Br J Anaesth 2002; 88: 621–4

[36] Fornage, B.D., *Peripheral nerves of the extremity: imaging with ultrasound*, Radiology 1988; 167: 179–82

[37] Graif, M., Seton, A., Nerubai, J., Horoszowski, H., Itzchak, Y., *Sciatic nerve: sonographic evaluation and anatomic-pathologic considerations*, Radiology 1991; 181: 405–8

[38] Kapral, S., Krafft, P., Eibenberger, K., Fitzgerald, R., Gosch, M., Weinstabl, C., *Ultrasound-guided supraclavicular approach for regional anesthesia of the brachial plexus*, Anesth Analg 1994; 78: 507–13

[39] Kirchmair, L., Entner, T., Wissel, J., Moriggl, B., Kapral, S., Mitterschiffthaler, G., *A study of the paravertebral anatomy for ultrasound-guided posterior lumbar plexus block*, Anesth Analg 2001; 93: 477–81

[40] Marhofer, P., Schrogendorfer, K., Koinig, H., Kapral, S., Weinstabl, C., Mayer, N., *Ultrasonographic guidance improves sensory block and onset time of three-in-one blocks*, Anesth Analg 1997; 85: 854–7

[41] Marhofer, P., Schrogendorfer, K., Andel, H., *Combined sciatic nerve—3 in 1 block in high risk patient*, Anaesthesiol Intensivmed Notfallmed Schmerzther 1998; 33: 399–401

[42] Marhofer, P., Schrogendorfer, K., Wallner, T., Koinig, H., Mayer, N., Kapral, S., *Ultrasonographic guidance reduces the amount of local anesthetic for 3-in-1 blocks*, Reg Anesth Pain Med 1998; 23: 584–8

[43] Grau, T., Leipold, R.W., Conradi, R., Martin, E., Motsch, J., *Efficacy of ultrasound imaging in obstetric epidural anesthesia*, J Clin Anesth 2002; 14: 169–75

[44] Marhofer, P., Greher, M., Sitzwohl, C., Kapral, S., *Ultrasonographic guidance for infraclavicular plexus anaesthesia in children*, Anaesthesia 2004; 59: 642–6

[45] Sandhu, N.S., Capal, L.M., *Ultrasound-guided infraclavicular brachial plexus block*, Br J Anaesth 2002; 89: 254–9

[46] Marhofer, P., Greher, M., Kapral, S., *Ultrasound guidance in regional anesthesia,* British Journal of Anaesthesia 94 (1): 7–17 (2005)

[47] Micromaxx : Portable diagnostic ultrasound overview, http://www.sonosite.com/products/micromaxx/, Accédé en Janvier 2008

[48] Shapiro, L., Stockman, G., *Computer Vision*, Prentice-Hall, Inc. 2001

[49] Hough, P.V.C., *Machine Analysis of Bubble Chamber Pictures*, Proc. Int. Conf. High Energy Accelerators and Instrumentation, 1959

[50] Duda, R. O., Hart, P.E., *Use of the Hough Transformation to Detect Lines and Curves in Pictures,* Comm. ACM, 1972, Vol. 15, pp. 11–15.

[51] Ballard, D.H., *Generalizing the Hough Transform to Detect Arbitrary Shapes*, Pattern Recognition, 1981, Vol.13, No.2, p.111-122.

[52] Hough Transform, http://en.wikipedia.org/wiki/Hough_transform/, Accédé en Janvier 2008

[53] Carter, P., *A brief history of GUI,* http://cognitivevent.com/gui_history.html, Accédé en Janvier 2008

[54] Runciman, W.B., Sellen, A., Webb, R.K., Williamson, J.A., Currie, M., Morgan, C., Russell, W.J., *The Australian Incident Monitoring Study*, Errors, incidents and accidents in anaesthetic practice. Anaesth.Intensive Care. 1993 Oct;21(5):506-19.

[55] Pierce, E.C., *Risk management in anesthesia. Introduction,* Int.Anesthesiol.Clin. 1989;27(3):133-6

[56] Gaba, D.M., Maxwell, M., DeAnda, A., *Anesthetic mishaps: breaking the chain of accident evolution*, Anesthesiology. 1987 May;66(5):670-6

[57] Zinser, K., Frischenschlager, F., *Multimedia's push into power*, Spectrum, IEEE 1994;31(7):44-8.

[58] Allnutt, M.F., *Human factors in accidents*, Br.J.Anaesth. 1987;(59):856-64

[59] Gurushanthaiah, K., Weinger, M.B., Englund, C.E., *Visual display format affects the ability of anesthesiologists to detect acute physiologic changes. A laboratory study employing a clinical display simulator,* Anesthesiology. 1995 Dec; 83(6):1184-93.

[60] Michels, P., Gravenstein, D., Westenskow, D.R., *An integrated graphic data display improves detection and identification of critical events during anesthesia*, J.Clin.Monit. 1997 Jul;13(4):249-59.

[61] Blike, G.T., Surgenor, S.D., Whalen, K., *A graphical object display improves anesthesiologists' performance on a simulated diagnostic task*, J.Clin.Monit.Comput. 1999 Jan;15(1):37-44.

[62] Maintz, J.B.A., Viergever, M.A., *A survey of medical image registration*, Med. Image Anal., 1998, vol. 2, no. 1, pp. 1–36.

[63] Dong, L., Pelle, G., Brun, P., Unser, M., *Model-based boundary detection in echocardiography using dynamic programming technique*, SPIE Image Proc. Conf., 1991, vol. 1445, pp. 178–187.

[64] Gustavsson, T., Abu-Gharbieh, R., Hamarneh, G., Liang, Q., *Implementation and comparison of four different boundary detection algorithms for quantitative ultrasonic measurements of the human carotid artery*, Proc. IEEE Comp. Cardiol., 1997, pp. 69–72.

[65] Gustavsson, T., Molander, S., Pascher, R., Liang, Q., Broman, H., Caidahl, K., *A model-based procedure for fully automated boundary detection and 3D reconstruction from 2D echocardiograms*, Proc. IEEE Comp. Card., Lund, Sweden, 1994, pp. 209–212.

[66] Paterni, V. G. M., Demi, M., Benassi, A., Gemignani, V., *A real-time contour tracking system to investigate the cross-area sectional changes of the aorta*, Computers in Cardiology, 2000, vol. 27, pp. 599–602.

[67] Bosch, J. G., Van Burken, G., Schukking, S. S., Wolff, R., Van de Goor, A. J., Reiber, J. H. C., *Real-time frame-to-frame automatic contour detection on echocardiograms*, Computers in Cardiology, pp. 29–32, 1994.

[68] Gemignani, V., Demi, M., Paterni, M., Benassi, A., *Real-time implementation of a new contour tracking procedure in a multi-processor DSP system*, Circ. Sys. Commun. Comp., pp. 3521–3526, 2000.

[69] Gee, A.H., Prager, R.W., *Sequential 3D diagnostic ultrasound using the Stradx algorithm*, Medical Image Computing and ComputerAssisted Intervention, C. Taylor and A. Colchester, Eds. New York:, Springer, 1999, pp. 716–725

[70] Abolmaesumi, P., Salcudean, S. E., Zhu, W.H., *Visual servoing for robot-assisted diagnostic ultrasound*, World Congr. on Medical Physics and Biomedical Engineering, 2000, vol. 4, Chicago, IL, pp. 2532-2535.

[71] Kanzanzides, P., Chang, J., Iordachita, I., Li, J., Ling, C.C., Fichtinger, G., *Design and validation of an image-guided robot for small animal research,* Medical Image Computing and Computer-Assisted Intervention, p50-57, 2006

[72] Waspe, A.C., Cakiroglu, H.J., Lacefield, J.C., Fenster, Q., *Design, calibration and evaluation of a robotic needle-positioning system for small animal imaging applications*, Physics in Medicine and Biology, 2007, 52(7):1863-1878.

[73] Fronheiser, M.P., Whitman, J., Ivancevich, N.M., Smith, S.W., *3-D Ultrasound Guidance of Surgical Robotic: Autonomous Guidance and Catheter Transducers*, IEEE Ultrasonics Symposium, 2007, p2527-2530.

[74] Ayadi, A., Bour, G., Aprahamian, M., Bayle, B., Graebling, P., Gangloff, J., Soler, L., Egly, J.M., Marescaux, J., *Fully Automated Image-Guided Needle Insertion: Application to small animal biopsies*, Proceedings of the 29th annual international conference of the IEEE EMBS, 2007, August, 194-197.

[75] Okazawa, S., Ebrahimi, R., Chuang, J., Salcudean, S.E., Rohling, R., *Hand-held steerable needle device*, IEEE/ASME transactions on mechatronics, 2005; 10(3): 285-296

ANNEXE 1 – CODE MATLAB

```
function varargout = houghGUI(varargin)
% Fichier M pour houghGUI.fig

% Dernière modification 30 janvier 2009

% Début du code d'initialisation – NE PAS MODIFIER
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @houghGUI_OpeningFcn, ...
                   'gui_OutputFcn',  @houghGUI_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% Fin du code d'initialisation – NE PAS MODIFIER
```

```
% --- Appel des paramètres d'initialisation

function houghGUI_OpeningFcn(hObject, eventdata, handles, varargin)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



handles.output = hObject;

guidata(hObject, handles);


% Initialisation des paramètres de base


S=[];

cylx=[];

cylz=[];

disp('Image buffer S initialized to []...');

imgcount=0;

disp('imgcount initialized to 0...');

firstx=0;

firsty=0;

trix=0;

triy=0;

disp('trix triy firstx firsty all initialized to 0...');

successful=0;

disp('successful is first set to false (0)...');

thresh=500;
```

```
disp('thresh is first initialized to 95...');

ultrasound=0;

disp('ultrasound is first set to 0...');

pos=0;

disp('position is first set to 0...');




% --- Les arguments sortants de cette fonction retournent à la ligne de commande

function varargout = houghGUI_OutputFcn(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1 MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11


varargout{1} = handles.output;




% --- Cette fonction s'execute quand le bouton Initialisation est pressé

function Initialisation_Callback(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1 MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11
```

```
% Initialise les servo-moteurs à leurs positions initiales, selon des distances % physiques prédéfinies du bras fantôme et du bras robotique.

% Les servo-moteurs prennent leurs positions initiales et attendront l'insertion % manuelle de la sonde d'ultrasons dans le bras-servo.


ultrasound=1;


Ser=serial('COM1', 'BaudRate',115200,'Terminator', 'CR/LF');

disp('Serial Port initialized');

fclose(Ser);

disp('Serial Port closed');

fopen (Ser);

disp('Serial Port opened');


% M0 est le moteur rotatif de base.

% Il participe au contrôle du mouvement de la sonde d'ultrasons.


M00='#0P0';

M0a='#0P1000 T10000';

M0b='#0P1300 T10000';

%M0c='#0P1400 T10000';

%M0d='#0P1200 T20000';


% M1 est le moteur d'élévation du premier segment.

% M0 et M1 reproduisent le mouvement rotatif et axial de l'épaule humaine


M10='#1P0';
```

```
M1a='#1P850 T10000';

% M2 est le moteur de rotation du second segment.
% Il reproduit le mouvement rotatif du coude humain.

M20='#2P0';
M2a='#2P1900 T10000';

%M12=strcat(M1a,' ',M2a);

% M3, M4 et M5 reproduisent le mouvement 3-D du poignet humain.

% M3 est le moteur de rotation dans le plan x-y

M30='#3P0';
M3a='#3P1400 T10000';
%M3b='#3P1500';

% M4 est le moteur de rotation dans le plan x-z

M40='#4P0';
M4a='#4P1000 T10000';
%M4b='#4P1500';

% M5 est le moteur de rotation dans le plan y-z

M50='#5P0';
M5a='#5P1200 T10000';
```

```
%M5b='#5P1500';


% M6 est le moteur qui contrôle le mouvement de la pince qui tient en place la
% sonde d'ultrasons.

M60='#6P0';
M6a='#6P750 T10000';
M6b='#6P1500 T5000';

% Par ressemblance au premier bras, le second bras robotique est composé des
% moteurs M7, M8 et M9, qui sont responsables des mouvements « épaule » (M7 et
% M8) et « coude » (M9).

M70='#7P0';
M7a='#7P1700 T10000';
M7b='#7P1200 T10000';

M80='#8P0';
M8a='#8P2200 T10000';
M8b='#8P1000 T10000';

M90='#9P0';
M9a='#9P1500 T10000';
M9b='#9P1600 T10000';

% Le second bras robotique se termine par un jeu de moteurs qui va positionner
% l'aiguille et le cathéter au-dessus du point d'insertion, les insérer en
même
```

```
% temps dans ce lieu, puis stopper l'aiguille une fois la veine percée et
% continuer à insérer le cathéter seul.

M100='#10P0';
M10a='#10P1500 T10000';
M10b='#10P750 T10000';

M110='#11P0';
M11a='#11P1500 T10000';
M11b='#11P2000 T10000';


%Tests pour des positions différentes

M0d='#0P1300 T10000';
M0e='#0P1300 T10000';
M0f='#0P1300 T10000';
M0g='#0P1300 T10000';
M0h='#0P1350 T10000';
M0i='#0P1350 T10000';


% M1 est le moteur d'élévation du premier segment.
% M0 et M1 reproduisent le mouvement rotatif et axial de l'épaule humaine

M1d='#1P750 T10000';
M1e='#1P750 T10000';
M1f='#1P950 T10000';
```

```
M1g='#1P950 T10000';

M1h='#1P750 T10000';

M1i='#1P850 T10000';


% M2 est le moteur de rotation du second segment.

% Il reproduit le mouvement rotatif du coude humain.


M2d='#2P2000 T10000';

M2e='#2P1900 T10000';

M2f='#2P1600 T10000';

M2g='#2P1700 T10000';

M2h='#2P2100 T10000';

M2i='#2P2100 T10000';


%M12=strcat(M1a,' ',M2a);


% M3, M4 et M5 reproduisent le mouvement 3-D du poignet humain.


% M3 est le moteur de rotation dans le plan x-y


M3d='#3P1350 T10000';

M3e='#3P1500 T10000';

M3f='#3P1700 T10000';

M3g='#3P1600 T10000';

M3h='#3P1200 T10000';

M3i='#3P1200 T10000';


% M4 est le moteur de rotation dans le plan x-z
```

```
M4d='#4P1000 T10000';
M4e='#4P1000 T10000';
M4f='#4P1020 T10000';
M4g='#4P1020 T10000';
M4h='#4P1050 T10000';
M4i='#4P1000 T10000';

% M5 est le moteur de rotation dans le plan y-z

M5d='#5P1050 T10000';
M5e='#5P1000 T10000';
M5f='#5P1220 T10000';
M5g='#5P1220 T10000';
M5h='#5P1150 T10000';
M5i='#5P1280 T10000';


% M6 est le moteur qui contrôle le mouvement de la pince qui tient en place la
% sonde d'ultrasons.

M60='#6P0';
M6a='#6P750 T10000';
M6b='#6P1500 T5000';


% Les commandes suivantes initialisent les bras robotiques.
% Le bras Arm 1 tient en place la sonde ultrasons.
```

```
% Le bras Arm 2 tient en place le groupe catheter/aiguille.
% MALLini initialise tous les moteurs.
% MALLarm1 envoie le bras Arm 1 en position initiale.
% MALLus1 envoie le bras Arm 1 en position de placement de la sonde d'ultrasons.
% MALLarm2 envoie le bras Arm 2 en position initiale.
% M0ALL envoie tous les moteurs en mode veille.

M0a
M1a
M2a
M3a
M4a
M5a
M6a

MALLini=strcat(M0a,M1a,M2a,M3a,M4a,M5a,M6a);

MALLarm1=strcat(M0b,M1a,M2a,M3a,M4a,M5a,M6a);
MALLus1=strcat(M0b,M1a,M2a,M3a,M4a,M5a,M6b);

MALLarm2=strcat(M7a,M8a,M9a,M10a,M11a);

M0ALL=strcat(M00,M10,M20,M30,M40,M50,M60,M70,M80,M90,M100,M110);

MALLmid1=strcat(M0d,M1d,M2d,M3d,M4d,M5d);
MALLmid2=strcat(M0e,M1e,M2e,M3e,M4e,M5e);
MALLfar1=strcat(M0f,M1f,M2f,M3f,M4f,M5f);
MALLfar2=strcat(M0g,M1g,M2g,M3g,M4g,M5g);
```

```
MALLclose1=strcat(M0h,M1h,M2h,M3h,M4h,M5h);
MALLclose2=strcat(M0i,M1i,M2i,M3i,M4i,M5i);



% Envoie les commandes d'initialisation aux moteurs, passant par le port série.

disp('MALLini');
fprintf(Ser,MALLini);
% fprintf(Ser,'#1P850T5000');
% fprintf(Ser,'#2P1900T5000');
% fprintf(Ser,'#3P1400T5000');
% fprintf(Ser,'#4P1000T5000');
% fprintf(Ser,'#5P1200T5000');
% fprintf(Ser,'#6P750T5000');
pause(3);


pause(3);

disp('MALLarm1');
fprintf(Ser,MALLarm1);
% fprintf(Ser,'#0P1300T5000');
pause(2);



% --- Cette fonction s'exécute quand le bouton Connect_Ultrasound est pressé.
```

```
function pbConnect_Ultrasound_Callback(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



% L'utilisateur doit placer la sonde ultrasons avant de presser sur le bouton

% Connect_Ultrasound. La connexion vidéo démarrera alors en temps réel.



disp('MALLus1');

fprintf(Ser,MALLus1);

pause(1)


disp('Connection to Ultrasound'); % Connexion établie!


% En connectant le TwinPact100 de Canopus à la machine d'ultrasons Micromaxx

% (à l'aide d'un câble S-video), puis celle-ci à l'ordinateur (à l'aide d'un

% câble Firewire, la vidéo apparaît une fois le bouton Connect_Ultrasound

% pressé. Il faut cependant s'assurer que le signal du TwinPact100 est
transmis

% en « Analog »; une transmission en « Digital » ou « RGB » résulterait en un

% écran noir au lieu de la vidéo.


obj = videoinput('winvideo', 1);
```

```
src_obj = getselectedsource(obj);

get(src_obj);

preview(obj);


disp('Connection successful');


% La connexion est établie!



% --- Cette fonction s'exécute quand le bouton Load est pressé.

function pbLoad_Callback(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



% Cette fonction permet d'ouvrir des images préenregistrées (au lieu de la
vidéo

% temps réel), et de les analyser pour en déterminer la veine.



%[filename, pathname] = uigetfile({'*.bmp';'*.jpg';'*.tif'});

%S = imread([pathname filename]);


%clear S;
```

```
if(ultrasound==1)

    disp('Loading snapshot');



    if(imgcount==0)
        disp('imgcount=0, first snapshot');
        S = getsnapshot(obj);
        imgcount = imgcount +1
        imgcount
        S={S}; % pour transformer la matrice en une cellule matricielle
        disp('moving position to next snapshot');
        fprintf(Ser,'#1P1000T5000');
        pause(1);
        fprintf(Ser,'#2P2000T5000 #3P1400T5000 #4P1100T5000');
        pause(2);
        fprintf(Ser,'#1P750T5000');
        pause(1);


        %fprintf(Ser,'#4P950T5000');
        %pause(1);
        %fprintf(Ser,'#2P1775T5000');
        %pause(1);
        %fprintf(Ser,'#1P1100T5000');


    else
        disp('next snapshot');
        S = [S; getsnapshot(obj)]; % pour concaténer une cellule matricielle
```

```
        imgcount
        imgcount = imgcount +1
        imgcount
        disp('moving position for additional snapshot');
%         fprintf(Ser,'#1P1000T5000');
%         pause(1);
%         fprintf(Ser,'#2P2000T5000 #3P1400T5000 #4P1100T5000');
%         pause(2);
%         fprintf(Ser,'#1P750T5000');
%         pause(1);

        %fprintf(Ser,'#4P900T5000');
        %pause(1);
        %fprintf(Ser,'#2P1750T5000');
        %pause(1);
        %fprintf(Ser,'#1P1100T5000');

    end




%     if(imgcount==0)
%         disp('imgcount=0, first snapshot');
%         S = getsnapshot(obj);
%         imgcount = imgcount +1
%         imgcount
%         S={S}; %make matrix a cell containing a matrix
%         disp('moving position to next snapshot');
```

```
%         fprintf(Ser,'#1P1050T5000');
%         pause(1);
%         fprintf(Ser,'#2P1850T5000');
%         % #3P1650T5000 #4P950T5000 #5P1300T5000');
%         pause(2);
%         fprintf(Ser,'#1P850T5000');
%         pause(1);
%
%         %fprintf(Ser,'#4P950T5000');
%         %pause(1);
%         %fprintf(Ser,'#2P1775T5000');
%         %pause(1);
%         %fprintf(Ser,'#1P1100T5000');
%
%     else
%         disp('next snapshot');
%         S = [S; getsnapshot(obj)]; %concatenate matrices in cell format
%         imgcount
%         imgcount = imgcount +1
%         imgcount
%         disp('moving position for additional snapshot');
%         fprintf(Ser,'#1P1000T5000');
%         pause(1);
%         fprintf(Ser,'#2P2000T5000 #3P1400T5000 #4P1100T5000');
%         pause(2);
%         fprintf(Ser,'#1P750T5000');
%         pause(1);
%
```

```
%          %fprintf(Ser,'#4P900T5000');
%          %pause(1);
%          %fprintf(Ser,'#2P1750T5000');
%          %pause(1);
%          %fprintf(Ser,'#1P1100T5000');
%
%      end


    % L'image fournie par la machine d'ultrasons contient une bordure.
    % Celle-ci doit être coupée pour accélérer l'algorithme d'analyse.
    % Pour Sonosite Micromaxx, les limites doivent être :
    % Coin supérieur gauche : X=94, Y=32
    % Coin inférieur droit : X=500, Y=200


    %UL=94; ul=32;
    %LR=500; lr=200;


    % Pour les tests qui ont été effectués au cours de ce projet, les limites
    % ont été fixées à :
    % Coin supérieur gauche : X=120, Y=75
    % Coin inférieur droit : X=550, Y=200

    % Ceci permet des calculs encore plus rapides, et résultats plus précis.
```

```
    UL=120; ul=75;
    LR=550; lr=200;

    disp('crop image');

    S{imgcount} = imcrop(S{imgcount},[UL ul LR lr]);
    handles.S{imgcount} = S{imgcount};
    axes(handles.axes1);
    imshow(S{imgcount});
    handles.output = hObject;
    guidata(hObject, handles);

    %S



    %axes(handles.axes4);
    %imshow(S); hold on;

    %figure; imshow(S);

else
    disp('click on browse image');
end



% --- Cette fonction s'exécute quand le bouton mid_position est pressé.
function mid_position_Callback(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
```

```
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



if(ultrasound==1)

    disp('positioning for first snapshot');
    fprintf(Ser, MALLmid1); % pour positionner le bras
    pause(20); % Il est necessaire de donner assez de temps de mouvement
    S = getsnapshot(obj); % pour prendre une image
    disp('snapshot taken')
    imgcount = imgcount +1 % pour incrementer le compteur d'images
    imgcount
    S={S}; % pour transformer la matrice en une cellule matricielle



    fprintf(Ser,'#1P950 T5000'); % pour lever le bras robotique
    pause(10);
    fprintf(Ser, MALLmid2); % pour positionner le bras
    disp('moving position to next snapshot');
    pause(20);
    S = [S; getsnapshot(obj)]; % pour concaténer une cellule matricielle
    disp('snapshot 2 taken')
    imgcount
    imgcount = imgcount +1
    imgcount
```

```
pos=1;



% L’image fournie par la machine d’ultrasons contient une bordure.
% Celle-ci doit être coupée pour accélérer l’algorithme d’analyse.
% Pour Sonosite Micromaxx, les limites doivent être :
% Coin supérieur gauche : X=94, Y=32
% Coin inférieur droit : X=500, Y=200



%UL=94; ul=32;
%LR=500; lr=200;



% Pour les tests qui ont été effectués au cours de ce projet, les limites
% ont été fixées à :
% Coin supérieur gauche : X=120, Y=75
% Coin inférieur droit : X=550, Y=200

% Ceci permet des calculs encore plus rapides, et résultats plus précis.



UL=120; ul=75;
LR=550; lr=200;

disp('crop image');
```

```
    % Les lignes de codes suivantes appliquent les coupures de bordures a
toutes
    % les images prises et les projette dans l'interface graphique.

    for n=1:imgcount
        S{imgcount} = imcrop(S{imgcount},[UL ul LR lr]);
        handles.S{imgcount} = S{imgcount};
        axes(handles.axes1);
        imshow(S{imgcount});
        handles.output = hObject;
        guidata(hObject, handles);
    end

    %S



    %axes(handles.axes4);
    %imshow(S); hold on;

    %figure; imshow(S);

else
    disp('click on browse image');
end

% --- Cette fonction s'exécute quand le bouton far_position est pressé.
```

```
function far_position_Callback(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



if(ultrasound==1)


    disp('positioning for first snapshot');

    fprintf(Ser, MALLfar1);

    pause(20);

    S = getsnapshot(obj);

    disp('snapshot taken')

    imgcount = imgcount +1

    imgcount

    S={S}; % pour transformer la matrice en une cellule matricielle



    fprintf(Ser,'#1P1100 T5000'); % pour lever le bras

    pause(10);

    fprintf(Ser, MALLfar2);

    disp('moving position to next snapshot');

    pause(20);

    S = [S; getsnapshot(obj)]; % pour concaténer une cellule matricielle

    disp('snapshot 2 taken')

    imgcount
```

```
imgcount = imgcount +1
imgcount

pos=2;



% L'image fournie par la machine d'ultrasons contient une bordure.
% Celle-ci doit être coupée pour accélérer l'algorithme d'analyse.
% Pour Sonosite Micromaxx, les limites doivent être :
% Coin supérieur gauche : X=94, Y=32
% Coin inférieur droit : X=500, Y=200


%UL=94; ul=32;
%LR=500; lr=200;


% Pour les tests qui ont été effectués au cours de ce projet, les limites
% ont été fixées à :
% Coin supérieur gauche : X=120, Y=75
% Coin inférieur droit : X=550, Y=200

% Ceci permet des calculs encore plus rapides, et résultats plus précis.


UL=120; ul=75;
LR=550; lr=200;
```

```
    disp('crop image');

    S{imgcount} = imcrop(S{imgcount},[UL ul LR lr]);
    handles.S{imgcount} = S{imgcount};
    axes(handles.axes1);
    imshow(S{imgcount});
    handles.output = hObject;
    guidata(hObject, handles);

    %S



    %axes(handles.axes4);
    %imshow(S); hold on;

    %figure; imshow(S);

else
    disp('click on browse image');
end



% --- Cette fonction s'exécute quand le bouton close_position est pressé.
function close_position_Callback(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
```

```
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11




if(ultrasound==1)

    disp('positioning for first snapshot');
    fprintf(Ser, MALLclose1);
    pause(20);
    S = getsnapshot(obj);
    disp('snapshot taken')
    imgcount = imgcount +1
    imgcount
    S={S}; % pour transformer la matrice en une cellule matricielle



    fprintf(Ser, MALLclose2);
    disp('moving position to next snapshot');
    pause(20);
    S = [S; getsnapshot(obj)]; % pour concaténer une cellule matricielle
    disp('snapshot 2 taken')
    imgcount
    imgcount = imgcount +1
    imgcount


    pos=3;
```

```
% L’image fournie par la machine d’ultrasons contient une bordure.
% Celle-ci doit être coupée pour accélérer l’algorithme d’analyse.
% Pour Sonosite Micromaxx, les limites doivent être :
% Coin supérieur gauche : X=94, Y=32
% Coin inférieur droit : X=500, Y=200


%UL=94; ul=32;
%LR=500; lr=200;


% Pour les tests qui ont été effectués au cours de ce projet, les limites
% ont été fixées à :
% Coin supérieur gauche : X=120, Y=75
% Coin inférieur droit : X=550, Y=200

% Ceci permet des calculs encore plus rapides, et résultats plus précis.


UL=120; ul=75;
LR=550; lr=200;

disp('crop image');

S{imgcount} = imcrop(S{imgcount},[UL ul LR lr]);
handles.S{imgcount} = S{imgcount};
axes(handles.axes1);
```

```
    imshow(S{imgcount});
    handles.output = hObject;
    guidata(hObject, handles);

    %S


    %axes(handles.axes4);
    %imshow(S); hold on;

    %figure; imshow(S);

else
    disp('click on browse image');
end



% --- Cette fonction s'exécute quand le bouton browse_image est pressé.
function browse_image_Callback(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11
```

```
[filename, pathname] = uigetfile({'*.tif';'*.bmp';'*.jpg'});



if(imgcount==0)
    S = imread([pathname filename]);
    imgcount = imgcount +1
    S={S}; %make matrix a cell containing a matrix
else
    S = [S; imread([pathname filename])];
    imgcount = imgcount +1
end



UL=200; ul=75;
LR=540; lr=180;

S{imgcount} = imcrop(S{imgcount},[UL ul LR lr]);
handles.S{imgcount} = S{imgcount};
axes(handles.axes1);
imshow(S{imgcount});
handles.output = hObject;
guidata(hObject, handles);

S



% --- Cette fonction s'exécute quand le bouton pbDetect est pressé.
function pbDetect_Callback(hObject, eventdata, handles)
```

```
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11


% Cette fonction détecte la veine dans chaque image prise.


if (ultrasound==1)
    disp('Arm1 back to initial position');
    fprintf(Ser,M6a);
    pause(10);
    fprintf(Ser,'#1P1200T10000');
    pause(5);
    fprintf(Ser,'#0P1000T10000');
    pause(5);
    %fprintf(Ser,MALLarm1);
    %pause(10);
    fprintf(Ser,'#2P1200T5000');
    pause(5);
    fprintf(Ser,'#1P750T5000');
    pause(5);


    disp('Arm1 all 0');
    fprintf(Ser,M0ALL);
end


disp('Detecting Vein');
```

```
for(cc=1:imgcount)

    imgcount

    cc

    handles.S{cc} = S{cc};

    axes(handles.axes1);

    %imshow(S{cc});

    handles.output = hObject;

    guidata(hObject, handles);


    disp('beginning analysis');


% Début de l'analyse


        % 1. Lecture de l'image

        I = handles.S{cc};


        % Conversion de l'image en noir et blanc

        Y = im2bw(256-I,0.75);

        % figure; imshow(Y);


        % Érosion de l'image. Technique utilisée pour éliminer le bruit, au

        % dépends d'une perte mineure d'information. Ceci accélère grandement

        % l'analyse.



        Y=medfilt2(Y);
```

```
%figure, imshow(Y)

se = strel('line',11,90);
I = imerode(Y,se);
%figure, imshow(I)

[y,x]=find(I);
[sy,sx]=size(I);

% 2. Recherche d'information pour appliquer la transformée de Hough.
totalpix = length(x);

% 3. Preallocation de mémoire pour la matrice de Hough.
HM = zeros(sy*sx,1);
R = get(handles.sldR,'value');
R2 = R^2;

% 4. Transformée de Hough

%%
% a. Préparation des matrices pour les calculs
b = 1:sy;
a = zeros(sy,totalpix);

y = repmat(y',[sy,1]);
x = repmat(x',[sy,1]);

b1 = repmat(b',[1,totalpix]);
```

```
b2 = b1;


%%
% b. Équation du cercle
a1 = (round(x - sqrt(R2 - (y - b1).^2)));
a2 = (round(x + sqrt(R2 - (y - b2).^2)));


%%
% c. Élimination des valeurs non-valides des matrices a et b.
b1 = b1(imag(a1)==0 & a1>0 & a1<sx);
a1 = a1(imag(a1)==0 & a1>0 & a1<sx);
b2 = b2(imag(a2)==0 & a2>0 & a2<sx);
a2 = a2(imag(a2)==0 & a2>0 & a2<sx);


ind1 = sub2ind([sy,sx],b1,a1);
ind2 = sub2ind([sy,sx],b2,a2);


ind = [ind1; ind2];


%%
% d. Reconstruction de la matrice de Hough
val = ones(length(ind),1);
data=accumarray(ind,val);
HM(1:length(data)) = data;
HM2 = reshape(HM,[sy,sx]);



% 5. Mise en évidence de la matrice de Hough dans l'interface.
```

```
axes(handles.axes2);
imshow(HM2,[]);



%%
% 6. Detection de la position du cercle de rayon R
[maxval, maxind] = max(max(HM2));
[B,A] = find(HM2==maxval);
axes(handles.axes3);
imshow(I); hold on;



mean(A), mean(B);



% Enregistrement des valeurs de centres de cercles dans des variables
% globales, pour utilisations futures lors des dessins des cercles

if(cc==1)

    cylx=mean(A);
    cylz=mean(B);
    cylx
    cylz

% Enregistrement des valeurs du premier centre de cercle détecté

    trix =mean(A);
```

```
            triy =mean(B);
            firstx=trix;
            firsty=triy;

        else

            % Sauvegarde des valeurs de centres de cercles subséquents dans
une
            % matrice globale.

            cylx=[cylx, mean(A)];
            cylz=[cylz, mean(B)];
            cylx
            cylz

          % Mesure de distance entre les valeurs de centre de cercle
précédente
          % trix et triy) et présente (mean(A) et mean (B)

          deltacenters=sqrt((mean(A) - trix)^2 + (mean(B) - triy)^2);
          deltacenters, thresh

      % Comparaison de cette différence à une valeur limite thresh

          if(deltacenters>thresh)
              successful=0;
              break;

          else
```

```
        successful=1;


    end
        trix =mean(A);
        triy =mean(B);
        trix
        triy
        cylx
        cylz

  end

% Dessin du cercle de centre trix, triy, autour de la veine détectée

  plot(trix,triy,'xr')



  axes(handles.axes4);
  hold off;
  imshow(handles.S{cc}); hold on;
  plot(trix,triy,'xr')

  t = 0:pi/20:2*pi;
  xdata = (trix+get(handles.sldR,'value').*cos(t))';
  ydata = (triy+get(handles.sldR,'value').*sin(t))';
  plot(xdata,ydata,'y:');
```

```
set(handles.txtX,'string',trix);
set(handles.txtY,'string',triy);
%set(handles.txtVal,'string',maxval);

% figure; imshow(handles.S{cc}); hold on;
plot(trix,triy,'xr')

t = 0:pi/20:2*pi;
xdata = (trix+get(handles.sldR,'value').*cos(t))';
ydata = (triy+get(handles.sldR,'value').*sin(t))';
plot(xdata,ydata,'y:');

% v=[trix; triy];



% figure; imshow(HM2,[]);
% figure; imshow(handles.S{cc});
% trix
% triy



%function write_file(filename)
%Function to write data to a file
%Arguments: filename , specified by user
%filename = 'test.txt';
%fid = fopen(filename, 'a');
%if (fid == -1)
%    error('cannot open file for writing');
```

```
        %end
%
        %fprintf(fid, '\n \n \n %f %f \n \n \n', trix, triy);

        %fclose(fid);


        %pause
end



successful

% Si la limite thresh n’est pas respectée, les centres de veines sont donc
% éloignés, et la veine n’est pas adéquate pour y insérer l’aiguille. La
valeur
% « successful » est donc 0.

if(successful==1)

    firstx
    firsty
    trix
    triy
    cylx
    cylz

%    fprintf(Ser,'#1P2000');
```

```
%    fclose(Ser);


end


% La valeur cyly est pré-déterminée
cyly(1)=16.8;
for (cc=1:imgcount-1)
    cyly(cc+1)=cyly(cc)-1.8;
    cyly(cc)
end
cyly(cc+1)


disp('ending analysis');


% --- Cette fonction s'exécute au déplacement du bouton glissant.
function sldR_Callback(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11


set(handles.txtR,'string',get(hObject,'Value'));



% --- Cette fonction s'exécute après le choix de paramètres complété.
function sldR_CreateFcn(hObject, eventdata, handles)
```

```
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11


if                                      isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end




function txtX_Callback(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11




% --- Cette fonction s’exécute après le choix de paramètres complété.
function txtX_CreateFcn(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
```

```
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11


if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function txtY_Callback(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



% --- Cette fonction s’exécute après le choix de paramètres complété.
function txtY_CreateFcn(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11
```

```
if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit4_Callback(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11
```

```
% --- Cette fonction s'exécute après le choix de paramètres complété.
function edit4_CreateFcn(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11
```

```
if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function txtalphaV_Callback(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



% --- Cette fonction s'exécute après le choix de paramètres complété.

function txtalphaV_CreateFcn(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end
```

```
function txtVal_Callback(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



% --- Cette fonction s'exécute après le choix de paramètres complété.

function txtVal_CreateFcn(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11


if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end



% --- Cette fonction s'exécute quand le bouton Position_Insertion_Box est

% pressé.
```

```
function Position_Insertion_Box_Callback(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



%beta=(180*asin(z/d))+(180*atan(h/s1));

%Pbeta=750+(beta/0.12);

%Sbeta=int2str(Pbeta);

%Posbeta=strcat('#1P',Sbeta);


% Paramètres initiaux:

s1=12.6; % Mesure du premier segment, OD

s2=10.6; % Mesure du second segment, DC (C étant une projection virtuelle de
C')

NL=4.5 %3.81; % Mesure de la longueur de l'aiguille

D=4.4; % Distance entre le point virtuel C et la base de l'aiguille B


% Valeurs de test : Angles horizontal et vertical, et coordonnées du point A

% (représentant le bout de l'aiguille)


alphaH=10; % Angle horizontal par défaut, en degrés

alphaV=30; % Angle vertical par défaut, en degrés

%Xa=18 - ((cylx(imgcount))/215); %17; %13.5

%Xa=18-(firstx/215);

Xa=16-(firstx/215);
```

```
% if pos==1
%     Ya=14;
% elseif pos==2
%     Ya=15 ;
% else
%     Ya=9.5 ;
% end

Ya=15;

%Za=-2.5 - ((cylz(imgcount))/200); %0;
Za=7.5-(firsty/200);

Xa
Ya
Za

% Les angles horizontaux et verticaux sont dépendants des pentes obtenues lors
% de la détection des coordonnées des centres de veines.

%slopeH=tan(pi*alphaH/180);
%slopeV=tan(pi*alphaV/180);

slopeH=0;
slopeV=0;
```

```
% for (cc=1:imgcount-1)
%    imgcount;
%    cc;
%    slopeh(cc)=(cyly(cc+1)-cyly(cc))/(cylx(cc+1)-cylx(cc));
%    slopeH=(slopeH + slopeh(cc))/2;
%    slopeh;
%    slopeH;
%     slopev(cc)=(cylz(cc+1)-cylz(cc))/(cyly(cc+1)-cyly(cc));
%     slopeV=(slopeV + slopev(cc))/2;
%    slopeV=slopeV+cylz(cc);
%    slopev
%    slopeV;
% end


cylx(2)
cylx(1)
cyly(2)
cyly(1)
cylz(2)
cylz(1)
%slopeH=(cyly(imgcount)-cyly(imgcount-1))/(cylx(imgcount)-cylx(imgcount-1));
slopeH=(cyly(2)-cyly(1))/((cylx(2)-cylx(1))/215);
alphaH=(180/pi)*atan(slopeH);


%slopeV=slopeV/imgcount;
%slopeV;
```

```
%slopeV=(cylz(imgcount)-cylz(imgcount-1))/(cylx(imgcount)-cylx(imgcount-1));
slopeV=((cylz(2)-cylz(1))/200)/(cyly(2)-cyly(1));
alphaV=(180/pi)*atan(slopeV);

slopeH
alphaH
slopeV
alphaV

set(handles.txtalphaH,'string',alphaH);
set(handles.txtalphaV,'string',alphaV);




% A partir du point A, on calcule le point B (base de l'aiguille) et le point
E
% (milieu de AB)
%     Xb=Xa+(-NL*cos(alphaV*pi/180)*cos((90-alphaH)*pi/180));
%     Yb=Ya+(-NL*cos(alphaV*pi/180)*sin((90-alphaH)*pi/180));

if alphaH < 0
    Xb=Xa+(-0.5*NL*cos(alphaV*pi/180)*sin((90+alphaH)*pi/180));
    Yb=Ya+(-NL*cos(alphaV*pi/180)*cos((90+alphaH)*pi/180));

else
    Xb=Xa+(+0.5*NL*cos(alphaV*pi/180)*sin((90-alphaH)*pi/180));
    Yb=Ya+(-NL*cos(alphaV*pi/180)*cos((90-alphaH)*pi/180));
end
```

```
%Zb=Za+(-NL*sin(alphaV*pi/180));
Zb=Za+(NL*sin(alphaV*pi/180));

Xb
Yb
Zb

%Xe=Xa+(-0.5*NL*cos(alphaV*pi/180)*cos(alphaH*pi/180));
%Ye=Ya+(-0.5*NL*cos(alphaV*pi/180)*sin(alphaH*pi/180));
%Ze=Za+(0.5*NL*sin(alphaV*pi/180));

% Le point virtuel C est l'intersection de l'axe du moteur 3 et l'axe du
moteur
% 4. C fait partie du bras robotique et forme un triangle avec le point O
% d'origine (0,0,0) et le point D (Xd, Yd, Zd), OD étant le premier segment
% s1 et DC le second segment s2.

% Ainsi, AB et AC sont perpendiculaires et nous pouvons donc utiliser la
% formule du produit scalaire AB.AC=0;
% xx'+yy'+zz'=0;
% (Xb-Xa)*Xc + (Yb-Ya)*Yc + (Zb-Za)*Zc - (Xb-Xa)*Xb - (Yb-Ya)*Yb - (Zb-Za)*Zb
=
% 0;

% Nous pouvons aussi trouver l'équation de la droite AB et utiliser l'équation
% de distance du point C a la droite AB :
% D=(aX+bY+cZ+d)/sqrt(a^2 + b^2 + c^2)
```

```
% L'équation de droite AB se trouve en solution au système d'équations à 3 inconnues pour les points A, B, et E, appartenant à la droite (AB : Z=aX+bY+c)

%M=[Xa Ya 1 ; Xb Yb 1 ; Xe Ye 1];
%N=[Za ; Zb ; Ze];
%R=M\N;

% L'équation de la droite (AB) serait donc:
% ABeq = Xc*R(1,1) + Yc*R(2,1) + Zc + R(3,1)
% En insérant cette équation dans l'équation de distance, on obtient :
% (D*sqrt((R(1,1))^2 + (R(2,1))^2 + 1^2)) - R(3,1) = Xc*R(1,1)+Yc*R(2,1)+Zc

% On peut donc calculer Xc= Xb-(D*((cos(alphaV*pi/180))*(cos(alphaH*pi/180))))

%Xc = Xb-(D*((cos(alphaV*pi/180))*(cos(alphaH*pi/180))));

% Ainsi, l'équation de distance, et celle du produit scalaire, sont résolues.

%O=[R(2,1) 1;(Yb-Ya) (Zb-Za)];
%P=[(D*sqrt(((R(1,1))^2)+((R(2,1))^2)+(1^2)))-R(3,1)-Xc*R(1,1);            (Xb-Xa)*Xb+(Yb-Ya)*Yb+(Zb-Za)*Zb-(Xb-Xa)*Xc];
%Q=O\P;

%Yc=Q(1,1);
%Zc=Q(2,1);

% Maintenant que les coordonnées du point C sont trouvées, on peut calculer
% les angles ODC et DOC dans le triangle O, D, C.
% d est la distance entre O et C
```

```
%d=sqrt((Xc^2)+(Yc^2)+(Zc^2));

%betaM8=180*(asin(-Zc/d))/pi; % Angle d'élévation du moteur M8

%anglexOC=(180*acos(Xc/d))/pi;
%betaM7=(180*acos((d^2 + s1^2 - s2^2)/(2*d*s1)))/pi; % Angle de rotation de
base

%angleM7=(180*acos(Yc/d))/pi;

%betaM9=(180*acos((-d^2 + s1^2 + s2^2)/(2*s2*s1)))/pi; % Angle ODC



% Les points B et C sont communs.
% Maintenant que les coordonnées du point B sont trouvées, on peut calculer
% les angles ODB et DOB dans le triangle O, D, B.
% d est la distance entre O et B

d=sqrt((Xb^2)+(Yb^2)+(Zb^2));

betaM8=(180*asin(Zb/d))/pi; % Angle d'élévation du moteur M8

angleyOB=(180*acos(Xb/d))/pi;

angleM7=(180*acos((d^2 + s1^2 - s2^2)/(2*d*s1)))/pi; % Angle de rotation de
base
anglexOB=(180*asin(Yb/d))/pi;
betaM7=90-(anglexOB + angleM7);
```

```
betaM9=(180*acos((-d^2 + s1^2 + s2^2)/(2*s2*s1)))/pi; % Angle ODB


%PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11

if(ultrasound==0)
    % Ouverture du port de communication COM1
    % Ceci permet d'envoyer les commandes aux moteurs du second bras robotique

    Ser=serial('COM1', 'BaudRate',115200,'Terminator', 'CR/LF');
    disp('Serial Port initialized');
    fclose(Ser);
    disp('Serial Port closed');
    fopen (Ser);
    disp('Serial Port opened');
end


% Calculs de position du moteur M8
PbetaM8=1970-(betaM8/0.12);
SbetaM8=int2str(PbetaM8);
PosbetaM8=strcat('#8P',SbetaM8,'T10000');
PbetaM8

% Calculs de position du moteur M9
PbetaM9=2150-((180-betaM9)/0.12);
SbetaM9=int2str(PbetaM9);
```

```
PosbetaM9=strcat('#9P',SbetaM9,'T10000');

PbetaM9


% Calculs de position du moteur M7

PbetaM7=1600-(angleM7/0.12);

SbetaM7=int2str(PbetaM7);

PosbetaM7=strcat('#7P',SbetaM7,'T10000');

PbetaM7


% Calculs de position du moteur M10


% if alphaH < 0

%     if betaM9 < 90

%          PbetaM10=1700 + ((90+alphaH)/0.12)-(angleM7/0.12)-((betaM9-90)/0.12)
+ 150;

%     else

%          PbetaM10=1700 + ((90+alphaH)/0.12)-(angleM7/0.12)-((90-betaM9)/0.12)
+ 150;

%     end

%

% else

%     if betaM9 < 90

%          PbetaM10=1700 - ((90-alphaH)/0.12)-(angleM7/0.12)-((betaM9-90)/0.12)
+ 150;

%     else

%          PbetaM10=1700 - ((90-alphaH)/0.12)-(angleM7/0.12)-((90-betaM9)/0.12)
+ 150;

%     end

%
```

```
% end


if alphaH < 0
    if betaM9 < 90
        PbetaM10=1700 + ((90+alphaH)/0.12)-(1600-PbetaM7)+(1250-PbetaM9) +65;
    else
        PbetaM10=1700 + ((90+alphaH)/0.12)-(1600-PbetaM7)+(PbetaM9-1250) +65;
    end


else
    if betaM9 < 90
        PbetaM10=1700 - ((90-alphaH)/0.12)-(1600-PbetaM7)+(1250-PbetaM9);
    else
        PbetaM10=1700 - ((90-alphaH)/0.12)-(1600-PbetaM7)+(PbetaM9-1250);
    end


end



alphaH
angleM7
PbetaM7
betaM7
betaM8
betaM9
PbetaM9


SbetaM10=int2str(PbetaM10);
```

```
PosbetaM10=strcat('#10P',SbetaM10,'T10000');
PbetaM10

% Calculs de position du moteur M11
if alphaV < 0
    PbetaM11=2100+(alphaV/0.12);
else
    PbetaM11=2100-(alphaV/0.12);
end

SbetaM11=int2str(PbetaM11);
PosbetaM11=strcat('#11P',SbetaM11,'T10000');



fprintf(Ser,MALLarm2);
pause(10);
fprintf(Ser,'#8P1900T10000');
pause(10);
fprintf(Ser,PosbetaM9);
pause(10);
fprintf(Ser,PosbetaM7);
pause(10);
fprintf(Ser,PosbetaM10);
pause(10);
fprintf(Ser,'#11P1500T10000');
pause(10);
fprintf(Ser,PosbetaM8);
pause(10);
```

```
% Positionnement du second bras robotique
%disp('Positionning Insertion Box');
%fprintf(Ser,MALLarm2);
%fprintf(Ser,M7b);
%pause(5);
%disp('Test insertion');
%fprintf(Ser,M9b);



% --- Cette fonction s'exécute quand le bouton Val_Insertion est pressé.
function Val_Insertion_Callback(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



% s1 et s2 sont les longueurs des segments en cm
% h est la difference de hauteur entre les segments 1 et 2
%
% s1=25;
% s2=17.5;
```

```
% h=4.6
%
% % d est la distance entre le point d'origine (0,0,0) et la destination
(x,y,z)
%
% d=sqrt(x^2 + y^2 + z^2);
%
% % beta est le mouvement angulaire requis pour les moteurs M1 et M2
% % Une multiplication par 180 est nécessaire pour convertir en radians
% % Les moteurs contiennent 1500 pas de position, capables de former des
angles
% % de 180 degrés. Donc, chaque mouvement de position affecte l'angle de 0.12
% % degrés (180/1500.
% % Aussi, il faut ajouter la valeur 750, qui est le positionnement du moteur
à
% % l'angle 0 degrés, afin de déterminer la position P.
%
% beta=(180*asin(z/d))+(180*atan(h/s1));
% Pbeta=750+(beta/0.12);
% Sbeta=int2str(Pbeta);
% Posbeta=strcat('#1P',Sbeta);
%
% Ser=serial('COM1', 'BaudRate',115200,'Terminator', 'CR/LF');
% fopen (Ser);
%
% fprintf(Ser,Posbeta);
%
%
%
```

```
% % alpha est le mouvement angulaire requis pour le moteur M0
% alpha=(180*acos((d^2 + s1^2 - s2^2)/(2*d*s1)))/pi;
%
% fprintf(Ser,'#1P2000');
%
%
% % phi est le mouvement angulaire requis pour le moteur M3
% phi=(180*acos((-d^2 + s1^2 + s2^2)/(2*s2*s1)))/pi;
%
% fprintf(Ser,'#1P2000');
%
%
%
% % theta est le mouvement angulaire requis pour le moteur M4
% theta=(180*acos((d^2 - s1^2 + s2^2)/(2*s2*d)))/pi;
%
% fprintf(Ser,'#1P2000');
%
%
% fprintf(Ser,'#1P2000');
%
% fclose(Ser);



fprintf(Ser,'#14P1500');
fprintf(Ser,'#15P1500')
pause(10);
```

```
fprintf(Ser,'#14P0');

fprintf(Ser,'#15P0');

fprintf(Ser,PosbetaM11);

pause(10);


fprintf(Ser,'#15P1500');

pause(15);


fprintf(Ser,'#15P0');



% --- Cette fonction s'exécute quand le bouton Info est pressé.

function pbInfo_Callback(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



web('http://newanesth.com/','-browser');







% --- Cette fonction s'exécute quand le bouton exit est pressé.

function exit_Callback(hObject, eventdata, handles)
```

```
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



fprintf(Ser,M0ALL);

fprintf(Ser,'#7P0');

fprintf(Ser,'#8P0');

fprintf(Ser,'#9P0');

fprintf(Ser,'#10P0');

fprintf(Ser,'#11P0');


fclose(Ser);


disp('Serial port closed')


%Clear all;

%Close all;




% --- Cette fonction s’exécute quand le bouton draw_vein est pressé.

function draw_vein_Callback(hObject, eventdata, handles)
```

```
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



% function [Cylinder EndPlate1 EndPlate2] =
Cylinder(X1,X2,r,n,cyl_color,closed,lines)
%
% Cette fonction construit un cylindre connectant deux centres
%
% Usage :
% [Cylinder EndPlate1 EndPlate2] = Cylinder(X1+20,X2,r,n,'r',closed,lines)
% Cylinder([20 10 05],[05 30 40],8,60,'b',0,1);
% Cylinder(X1,X2,r,n,'r',closed,lines);
%
% Cylinder-------Cylindre
% EndPlate1------Plan de départ
% EndPlate2------Plan de fin
% X1 et X2 forment les vecteurs 3x1 des deux points de centre
% r est le rayon du cylindre
% n est le nombre d'éléments sur la circonférence du cylindre (détails)
% cyl_color définit les couleurs du cylindre (ex. 'r','b',[0.52 0.52 0.52])
% closed=1 pour un cylindre fermé
% closed=0 pour un cylindre ouvert
% lines=1 pour faire apparaître les segments de ligne sur la surface
```

```
% lines=0 pour ne faire apparaître que la surface du cylindre
%
% Entrées typiques pour cette fonction :
% X1=[10 10 10];
% X2=[35 20 40];
% r=1;
% n=20;
% cyl_color='b';
% closed=1;
%


%cyly(1)=0; %14;
%cyly(2)=100; %14.5;
%cyly(3)=200; %15;
%cyly(4)=300; %15.5;
%cyly(5)=400;




for(cc=1:imgcount-1)

    imgcount
    cc
    X1=[cyly(cc) ((cylz(cc))/200) ((cylx(cc))/215)];
    X2=[cyly(cc+1) ((cylz(cc+1))/200) ((cylx(cc+1))/215)];


    %X1=[cyly(cc) ((cylz(cc))/200) ((cylx(cc))/215)];
    %X2=[cyly(cc+1) ((cylz(cc+1))/200) ((cylx(cc+1))/215)];
```

```
%X1=[180 321 100];
%X2=[255 411 200];
%X1=[255 411 200];
%X2=[326 499 320];
r=30/200;
n=20;
if cc==1
    cyl_color='g';
else if cc==2
        cyl_color='b';
    else if cc==3
            cyl_color='r';
        else
            cyl_color='g';
        end
    end
end
closed=1;
lines=1;

axes(handles.axes5);

% Pour calculer la longueur du cylindre :
length_cyl=norm(X2-X1);

% Pour créer le cercle dans le plan YZ
```

```
t=linspace(0,2*pi,n)';
x2=r*cos(t);
x3=r*sin(t);

% Pour créer les points dans la direction X
x1=[0 length_cyl];

% Pour créer les points de cylindre dans la direction X
xx1=repmat(x1,length(x2),1);
xx2=repmat(x2,1,2);
xx3=repmat(x3,1,2);

% Pour dessiner deux cercles formant les bases du cylindre
if closed==1
    hold on
    EndPlate1=fill3(xx1(:,1),xx2(:,1),xx3(:,1),'r');
    EndPlate2=fill3(xx1(:,2),xx2(:,2),xx3(:,2),'r');
end

% Pour dessiner le cylindre dans la direction X avec la longueur requise
% à partir de l'origine
Cylinder=mesh(xx1,xx2,xx3);

% Pour définir le vecteur unitaire dans la direction X
unit_Vx=[1 0 0];

% Pour calculer l'angle entre la direction X et la direction désirée
% L'utilisation du produit scalaire est requise
```

```
    angle_X1X2=acos(  dot(  unit_Vx,(X2-X1)  )/(  norm(unit_Vx)*norm(X2-X1))
)*180/pi;

    % Pour trouver l'axe de rotation du cylindre
    axis_rot=cross([1 0 0],(X2-X1) );

    % Pour placer le cylindre et ses bases dans le bon plan X, Y, Z
    if angle_X1X2~=0 % Le cylindre est dans le plan X, donc pas de rotation
        rotate(Cylinder,axis_rot,angle_X1X2,[0 0 0])
        if closed==1
            rotate(EndPlate1,axis_rot,angle_X1X2,[0 0 0])
            rotate(EndPlate2,axis_rot,angle_X1X2,[0 0 0])
        end
    end

    % Positionnement du cylindre au bon point de depart dans le plan X, Y, Z
    if closed==1
        set(EndPlate1,'XData',get(EndPlate1,'XData')+X1(1))
        set(EndPlate1,'YData',get(EndPlate1,'YData')+X1(2))
        set(EndPlate1,'ZData',get(EndPlate1,'ZData')+X1(3))

        set(EndPlate2,'XData',get(EndPlate2,'XData')+X1(1))
        set(EndPlate2,'YData',get(EndPlate2,'YData')+X1(2))
        set(EndPlate2,'ZData',get(EndPlate2,'ZData')+X1(3))
    end
    set(Cylinder,'XData',get(Cylinder,'XData')+X1(1))
    set(Cylinder,'YData',get(Cylinder,'YData')+X1(2))
    set(Cylinder,'ZData',get(Cylinder,'ZData')+X1(3))
```

```
    % Pour configurer la couleur du cylindre et de ses bases
    set(Cylinder,'FaceColor',cyl_color)
    if closed==1
        set([EndPlate1 EndPlate2],'FaceColor',cyl_color)
    else
        EndPlate1=[];
        EndPlate2=[];
    end

    % Pour enlever les lignes
    if lines==0
        set(Cylinder,'EdgeAlpha',0)
    end
end




for(cc=1:imgcount-1)

    imgcount
    cc
    X1=[((cylx(cc))/215) cyly(cc) ((cylz(cc))/200)];
    X2=[((cylx(cc+1))/215) cyly(cc+1) ((cylz(cc+1))/200)];

    %X1=[cyly(cc) ((cylz(cc))/200) ((cylx(cc))/215)];
    %X2=[cyly(cc+1) ((cylz(cc+1))/200) ((cylx(cc+1))/215)];
```

```
%X1=[180 321 100];
%X2=[255 411 200];
%X1=[255 411 200];
%X2=[326 499 320];
r=30/200;
n=20;
if cc==1
    cyl_color='g';
else if cc==2
        cyl_color='b';
    else if cc==3
            cyl_color='r';
        else
            cyl_color='g';
        end
    end
end
closed=1;
lines=1;

axes(handles.axes6);

% Pour calculer la longueur du cylindre :
length_cyl=norm(X2-X1);

% Pour créer le cercle dans le plan YZ
t=linspace(0,2*pi,n)';
x2=r*cos(t);
```

```
    x3=r*sin(t);

    % Pour créer les points dans la direction X
    x1=[0 length_cyl];

    % Pour créer les points de cylindre dans la direction X
    xx1=repmat(x1,length(x2),1);
    xx2=repmat(x2,1,2);
    xx3=repmat(x3,1,2);

    % Pour dessiner deux cercles formant les bases du cylindre
    if closed==1
        hold on
        EndPlate1=fill3(xx1(:,1),xx2(:,1),xx3(:,1),'r');
        EndPlate2=fill3(xx1(:,2),xx2(:,2),xx3(:,2),'r');
    end

    % Pour dessiner le cylindre dans la direction X avec la longueur requise
    % à partir de l'origine
    Cylinder=mesh(xx1,xx2,xx3);

    % Pour définir le vecteur unitaire dans la direction X
    unit_Vx=[1 0 0];

    % Pour calculer l'angle entre la direction X et la direction désirée
    % L'utilisation du produit scalaire est requise
    angle_X1X2=acos(  dot(  unit_Vx,(X2-X1)  )/(  norm(unit_Vx)*norm(X2-X1))
)*180/pi;
```

```
% Pour trouver l'axe de rotation du cylindre
axis_rot=cross([1 0 0],(X2-X1) );

% Pour placer le cylindre et ses bases dans le bon plan X, Y, Z
if angle_X1X2~=0 % Rotation is not needed if required direction is along X
    rotate(Cylinder,axis_rot,angle_X1X2,[0 0 0])
    if closed==1
        rotate(EndPlate1,axis_rot,angle_X1X2,[0 0 0])
        rotate(EndPlate2,axis_rot,angle_X1X2,[0 0 0])
    end
end

% Positionnement du cylindre au bon point de depart dans le plan X, Y, Z
if closed==1
    set(EndPlate1,'XData',get(EndPlate1,'XData')+X1(1))
    set(EndPlate1,'YData',get(EndPlate1,'YData')+X1(2))
    set(EndPlate1,'ZData',get(EndPlate1,'ZData')+X1(3))

    set(EndPlate2,'XData',get(EndPlate2,'XData')+X1(1))
    set(EndPlate2,'YData',get(EndPlate2,'YData')+X1(2))
    set(EndPlate2,'ZData',get(EndPlate2,'ZData')+X1(3))
end
set(Cylinder,'XData',get(Cylinder,'XData')+X1(1))
set(Cylinder,'YData',get(Cylinder,'YData')+X1(2))
set(Cylinder,'ZData',get(Cylinder,'ZData')+X1(3))

% Pour configurer la couleur du cylindre et de ses bases
set(Cylinder,'FaceColor',cyl_color)
```

```
    if closed==1
        set([EndPlate1 EndPlate2],'FaceColor',cyl_color)
    else
        EndPlate1=[];
        EndPlate2=[];
    end

    % Pour enlever les lignes
    if lines==0
        set(Cylinder,'EdgeAlpha',0)
    end
end
```

```
function txtalphaH_Callback(hObject, eventdata, handles)
global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1 MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11
```

```
% --- Cette fonction s'exécute après le choix de paramètres complété.

function txtalphaH_CreateFcn(hObject, eventdata, handles)

global trix triy cylx cyly cylz S imgcount firstx firsty successful thresh
ultrasound obj Ser M00 M0a M0b M0c M0d M10 M1a M20 M2a M30 M3a M40 M4a M50 M5a
M60 M6a M6b M70 M7a M7b M80 M8a M8b M90 M9a M9b M100 M10a M10b M110 M11a M11b
MALLini MALLarm1 MALLus1 MALLarm2 M0ALL M0d M1d M2d M3d M4d M5d M0e M1e M2e
M3e M4e M5e M0f M1f M2f M3f M4f M5f M0g M1g M2g M3g M4g M5g M0h M1h M2h M3h
M4h M5h M0i M1i M2i M3i M4i M5i MALLmid1 MALLmid2 MALLfar1 MALLfar2 MALLclose1
MALLclose2 pos PosbetaM7 PosbetaM8 PosbetaM9 PosbetaM10 PosbetaM11



if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end
```